TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Rasmus Tomsen 191983IAPM

IN-DEPTH FEATURE SELECTION ANALYSIS FOR THE IOT BASED BOTNET ATTACK DETECTION

Master's thesis

Supervisor: Sven Nõmm
PhD

Hayretdin Bahşi
PhD

Tallinn 2021

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Rasmus Tomsen 191983IAPM

PÕHJALIK TUNNUSTE ARVUTAMISE ANALÜÜS ASJADE INTERNETI ZOMBIVÕRGU RÜNNAKUTE AVASTAMISEKS

Magistritöö

Juhendaja: Sven Nõmm
PhD

Hayretdin Bahşi
PhD

Tallinn 2021

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Rasmus Tomsen

07.05.2021

Abstract

The current thesis analyses the use of feature selection in unsupervised anomaly detection and supervised classification algorithms to create accurate and efficient botnet detection models for small, resource constraint Internet of Things devices. The work analyses how applying distinct feature selection algorithms to different phases of an IoT device lifecycle affects the outcome of the learning models, if it is possible to create effective models by training a single model for multiple similar devices and which feature selection and botnet detection models provide the overall best results.

As a result of the thesis, thousands of different botnet detection models are created and evaluated on the data of 9 separate IoT devices. The pitched analysis goals are then accomplished based on the outcomes of the created models. Additionally, for any future analysis, the thesis also provides access to the full results of the models.

This thesis is written in English and is 83 pages long, including 7 chapters, 19 figures and 67 tables.

Annotatsioon

Käesolev magistritöö keskendub Asjade Interneti, ehk IoT pahavara tuvastamise analüüsile, kasutades selleks erinevaid nii juhendamisega kui ka juhendamiseta masinõppe algoritme. Põhirõhk töös on tunnuste arvutamisel, et kahandada olemasoleva andmestiku dimensionaalsust ning selle abil luua tõhusamaid mudeleid, mida saaks kasutada väikeste ning piiratud ressursiga IoT seadmete jaoks. Töö analüüsib, kuidas tunnuste vähendamine erinevatel IoT seadmete eluea etappidel muudab mudelite tulemusi. Lisaks sellele uurib töö, kas mitme sarnase IoT seadme jaoks on võimalik luua tõhusat kombineeritud masinõppe mudelit, et vähendada hooldusele kuluvat ressurssi ning millised tunnuste arvutamise algoritmid ja juhendamisega ning juhendamiseta masinõppe algoritmid annavad parimaid tulemusi IoT seadmete pahavara tuvastamisel.

Töö tulemusena valmisid tuhanded erinevad masinõppe baasil toimivad IoT zombivõrgu tuvastamismudelid, mille efektiivsust kontrolliti 9 erineva IoT seadme andmestiku peal. Mudelitest saadud tulemuste põhjal koostati laialdane analüüs, et leida vastused eelnevalt väljatoodud probleemidele. Lisaks korraldatud analüüsile, annab töö ligipääsu kõikide mudelite tulemustele, mille põhjal võib koostada lisaanalüüse tulevikus.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 83 leheküljel, 7 peatükki, 19 joonist, 67 tabelit.

List of abbreviations and terms

Ack	TCP acknowledgement message flooding based attack
COMBO	Botnet attack that consists of sending spam data and opening a connection to a specified IP address and port
DDoS	*Distributed Denial of Service*, attack type
IF	*Isolation Forest*, unsupervised learning algorithm
IoT	*Internet of Things*, network of devices
Junk	Spam data based botnet attack
KNN	*K-Nearest Neighbor*, supervised learning algorithm
LOF	*Local Outlier Factor*, supervised learning algorithm
ML	*Machine Learning*, study of computer algorithms
MLP	*Multi-Layer Perceptron*, supervised learning algorithm
Scan	Botnet attack to scan a network for vulnerable devices
SVM	*Support Vector Machine*, supervised learning algorithm
Syn	TCP connection establishing message flooding based attack
TCP	*Transmission Control Protocol* flooding based botnet attack
UDP	*User Datagram Protocol* flooding based botnet attack
UDPPLain	*User Datagram Protocol* flooding based botnet attack with fewer options

Table of contents

List of figures

List of tables

1 Introduction

In the last decade, the world of Internet of Things or IoT has grown rapidly, and it shows no signs of slowing down. The concept of IoT is having ordinary common everyday items and devices connected to each other locally or more often via the internet to make their normal usage more effective. These, often so-called “smart” devices can then collect and broadcast data from their surrounding environment or receive remote commands to then act upon. These kinds of devices could be anything, such as different security cameras, baby monitors, light switches, cars and so on [1]. The number of IoT devices has been growing exponentially and in 2020 the worldwide number of IoT connections surpassed the number of non-IoT devices, such as smartphones and personal computers for the first time in history with approximately 11.7 billion active IoT devices [2].

1.1 Motivation

Because of the ever-growing market, IoT devices have become targets of malware attacks to then use them for a variety of malicious activities. One of the most common such use-case is using the network of infected devices, called a botnet, for Distributed Denial of Service (DDoS) attacks to try to interrupt a targeted server or service normal workflow by overflooding the target with a large amount of network requests [3]. When previously it was more difficult to gain access and use many computers and devices to carry out DDoS attacks, then with the growing popularity of IoT devices, new wave of such attacks emerged. Using IoT devices instead to flood targets with packets meant that attackers had many smaller and more easily accessible devices available. Due to the sheer number of IoT devices the existing attack detection systems proved to be less effective than previously. The sent requests often found their way around these systems and the attacks were more difficult to handle. In addition to that, IoT devices are often less protected with weaker passwords and security countermeasures making them easier targets to infect with malware and therefore gain access to [4].

To deal with the novel issue, new systems had to be invented to be able to detect malicious IoT activity from normal one as early as possible. Although a lot of research has been put into creating signature-based algorithms that are able to detect just that, these kinds of models can often create many false positives due to the nature of the methods. Additionally, because of the sizes of IoT networks, maintaining the rule updates for the signature-based methods can prove to be difficult [5]. More recently, a lot more research has been put into Machine Learning based models. However, these models do not often consider the very limited resources IoT devices usually have. This means that these created models usually just try to create as accurate models as possible without paying too much attention to, for example, reducing the dimensionality of the data or using shallower learning methods [6], [7].

1.2 Problem statement

The main issue the thesis is focused on, is reducing the complexity of botnet detection models to make them more suitable for small IoT devices. This can further be divided into multiple sub-problems that the work aims to solve.

First, one of the biggest points is reducing the dimensionality of the data used in the detection models. In addition to the higher computational costs high dimensional data might cause, this kind of data can also create a problem called “curse of dimensionality”, where higher dimensional data becomes more sparse and much more data will be needed to get dependable results [6], [8]. The study plans to explore the novel idea of using feature selection methods to reduce the training and testing data feature sets and therefore optimise the learning models.

Secondly, while using complex deep learning or overall supervised learning models have proven to effective in detecting botnets, these models require previously obtained attack data to be present during training. This can be an issue with IoT devices that have not come into any prior contact with attack data or just have not been able to correctly label the attack data they have previously encountered. While research papers where anomaly detection models were used for IoT botnet detection do exist [9] [6], then the current paper extends the research of the topic further.

Combining the previous two issues, it is unclear if and how much does having prior knowledge of attack data affects the feature selection process for different detection models. The work aims to compare the effectiveness of feature selection when applied to different stages of IoT device's lifecycle of when a device has never encountered attack data before compared to a device which has already been infected with a malware.

Next, most of the previous work on similar subjects have usually created separate models for different devices. However, in a large network of IoT devices this can severely cause the maintenance and computation costs to grow. The thesis additionally tries to see if creating common models for multiple devices, that have a similar purpose (e.g., all cameras), could reduce those costs.

Lastly, a large variety of different feature selection, anomaly detection and classification algorithms have been proposed throughout the years that with slight modifications could potentially be used for IoT botnet detection as well. However, when dealing with IoT data, it is not often clear which methods would be able to achieve the highest accuracies in predicting attack data while consuming the least amount of computational power.

To sum up the pitched problems, the thesis aims to solve the following:

- Not enough previous research done on the topic of feature selection to reduce the dimensionality of the data for IoT botnet detection systems.
- The previous studies have not covered how applying feature selection to different parts of an IoT device's lifecycle can affect the outcome of feature selection and the overall botnet detection models.
- Most of the previous solutions have created a separate detection model for each different IoT device and the idea of lowering the maintenance costs by training a single model for multiple devices has not been researched enough.
- While different feature selection and learning models do exist, not enough has been done to see which ones work the best and most efficiently for IoT botnet detection.

1.3 Related work

A decent amount of research has already been done in the field of botnet detection using machine learning. Different specification-based [10], unsupervised anomaly-based [11],

supervised classification [12] [13] and deep learning [14] [15] ideas have been proposed that often produce highly accurate results [5]. However, most of such models cannot be directly adapted for different resource-constraint IoT devices and often work only on higher end devices.

Research on IoT based botnets has started to come out mostly only in the recent years. As IoT networks continue to grow, the botnets based on them continue to develop as well and more work needs to be done to detect and act upon the attacks. Different studies have been made to create models dedicated to finding anomalies in the normal workflow of an IoT device. These models are often developed from the previously mentioned and proven botnet detection ideas.

In the work proposed by [16], different supervised classification models and one neural network model were tested to detect between benign records and records sent from devices infected with Mirai malware. The detection models were built upon a middlebox device which all IoT devices in the network sent data through. While the models were able to detect Mirai malware with high accuracy, not much attention was paid to performance and computational costs as the models were not built on the IoT devices themselves. Additionally, only supervised methods were analysed, meaning that already labelled data was needed.

While [16], as well as most other studies are based on being able to differentiate between normal and attack records that potentially already infected devices send out, then ideas have been proposed on how to detect possible attacks as early as possible in the propagation and Command and Control server phases. In the work [17], a lightweight deep-packet anomaly detection model was created that is suitable for low-performance IoT devices as well. The work used bit-pattern matching for feature selection and a binary representation of features in a look-up table that is used for detecting anomalies in the data. The method was tested against a variety of different types of attacks and was able to achieve highly accurate results.

However, as suggested in [9], these early life detection models alone might become insufficient in the future, as early life detection datasets are difficult to acquire and simulate. Because of this, both the current thesis and the research paper [9] instead focus on already infected IoT devices and classification of the data they send out. The work set

up a test environment of 9 different commercial IoT devices from which three types of data was collected. In addition to normal data, the devices were then infected with BASHLITE and Mirai malware respectively and data was collected from the infected devices as well. The paper also made the dataset publicly available [18] and it will be used in the current thesis as well. The work then consisted of training and testing an experimental deep learning model, as well as three unsupervised anomaly detectors. While the models proved to be promising, the work did not center on optimization of the models to use them effectively on smaller IoT devices [9].

In addition to those, other works have been proposed to use different methods to effectively be able to detect possible IoT malware attacks. Such as [19], which proposed detecting Mirai attacks using neural networks and word embedding. Depending on the attack type, the created LSTM-RNN and BLSTM-RNN models were able to distinguish 98%-99% of the attack data. Other works have proposed for example a hybrid anomaly- and specification-based intrusion detection system based on the MapReduce architecture [20], a SIEM-based IoT DDoS attack detection system [21] as well as others [22], [23]. However, while many different approaches have been proposed, most of the works pay little to no attention to the low computation power of IoT devices and reducing the performance costs.

A few papers on the topic of IoT botnet detection and feature selection have been published in TalTech as well. Works proposed by [7] and [6] started investigating the usage of feature selection methods to reduce the number of features for the training and testing data and still provide accurate supervised and unsupervised anomaly detection models. The paper [6] tested some of the more common unsupervised learning models One-Class SVM and Isolation Forest with reduced datasets and [7] experimented on the supervised k-NN and Decision tree models. Both research papers showed much promising results and the current thesis will be largely based on those papers and will expand on the ideas proposed there by analysing and comparing a much larger number of different models.

1.4 Goals

The main goal of the thesis is to research the novel idea of using feature selection for IoT device attack data recognition and the accuracy and training and testing performance of

such models. In total, the goal can be divided into multiple sub-goals to solve the aforementioned problems.

First, the work will analyse a large number of different anomaly detection models in order to see how much does having prior knowledge of attack data affect the feature selection results and the performance of the attack recognition models. Since most IoT devices used in different applications have never encountered any malicious data before, then initially the feature selection and training of anomaly detection models for such devices is going to be made using normal data only. To see if these kinds of models would be able to detect attack records better if they had any prior attack data available during feature selection as well, then additional models will be created where different types of attack data is added to the feature selection dataset. Initially only one type of attack data, Gafgyt, is used in feature selection to see if that would help in predicting other types of malicious records as well. Later, these results are compared with models where Mirai data was available during training as well. Knowing if and how much does attack data affect the feature selection is especially useful for unsupervised anomaly detection models where the training will be done on only benign data regardless of the data from which features were selected from.

Additionally, the paper researches if it is possible to reduce the overall maintenance costs of IoT anomaly detection models by training the models on multiple devices of a similar category and creating a common model for those devices. Most previous works done on similar IoT botnet detection topics, such as [9] usually create a separate model for each device. While models like this can possibly have the advantage of reducing incorrect predictions caused by largely different data from different devices, they can suffer from the performance part of having to train and maintain many models.

While comparing the feature selection datasets and devices the models get trained on are the two most important problems proposed by the thesis, the paper also proposes a slightly smaller third goal of comparing other possible parameters of the learning models and finding the best performing values. Here, different feature selection algorithms, powers of feature sets and anomaly detection and classification methods are compared.

All the goals will be carried out on both the supervised and unsupervised malicious data detection models.

2 Analysis methods

An overview of different methods and data used in the analysis to solve the proposed goals is summarized here.

2.1 Data

The data which the current thesis analysis is based on was acquired from 9 real commercial IoT devices from an authentic test environment. The tests were performed by a group of researchers from Ben-Gurion University of the Negev and Singapore University of Technology and Design [18]. The devices the data was acquired from are the following:

- SimpleHome XCS7-1002-WHT security camera
- SimpleHome XCS7-1003-WHT security camera
- Provision PT-737E security camera
- Provision PT-838 security camera
- Samsung SNH-1011N web camera
- Philips B120N/10 baby monitor
- Ennio doorbell
- Danmini doorbell
- Ecobee thermostat

From each device, thousands of 115-feature benign records were collected. In addition to that, the devices were then infected by two of the most common botnet malware types, BASHLITE (referenced to as Gafgyt in most of the following paper) and Mirai and additional data was extracted from those as well. The attack data can also be divided into 10 different attack types,

such as Scan, Junk, UDP, TCP and COMBO for Gafgyt and Scan, Ack, Syn, UDP and UDPplain for Mirai [9].

Each record from the collected datasets consists of 115 features. The features can be described by three parameters: stream aggregation, time-frame and the statistics extracted from the packet stream. For stream aggregation the following values are available [18]:

- Source IP (*H*)
- Source MAC-IP (*MI*)
- Channel, or the source and destination IP (*HH*)
- Channel jitter, or the source and destination traffic jitter (*HH_jit*)
- Socket, or the source and destination IP and port (*HpHp*)

The time-frame is displayed as the decay factor lambda used in the damped window with values of L5, L3, L1, L0.1 and L0.01 which correspond to time windows of 100ms, 500ms, 1.5s, 10s and 1min respectively [24].

Lastly, the following statistics are collected from the packet stream [18]:

- Weight of the stream (*weight*)
- Mean of the stream (*mean*)
- Std. of the stream (*std*)
- Root squared sum of two stream variances (*radius*)
- Root squared sum of two stream means (*magnitude*)
- Approximated covariance between two streams (*cov*)
- Correlation coefficient of two streams (*pcc*)

For the thesis analysis, a random sample of data is selected from each generated dataset. Depending on the tests, the amount of data records that is gathered from each set varies.

However, when a test is dealing with either Gafgyt or Mirai data, then approximately an equal amount of each previously described attack type is chosen.

2.2 Feature selection datasets

For the first main goal of the thesis of seeing how much prior knowledge of malicious data affects the feature selection, the incoming data must be grouped into multiple different subsets. These subsets are then later used in both the unsupervised anomaly detection and the supervised classification analysis during the feature selection part. The sample sets are divided as following:

- benign – A data sample set that consists of only benign data. Usually in the size of 20000 benign records
- benign-gafgyt-balanced – A data sample set that consists of 50% benign and 50% Gafgyt data. Usually in the size of 10000 benign and 10000 Gafgyt records
- benign-gafgyt-unbalanced – A data sample set biased towards Gafgyt data that consists of about 10% benign and 90% Gafgyt data. Usually in the size of 2000 benign and 20000 Gafgyt records
- benign-gafgyt-mirai-balanced – A data sample set that consists of about 33.3% benign, 33.3% Gafgyt and 33.3% Mirai data. Usually in the size of 10000 benign, 10000 Gafgyt and 10000 Mirai records
- benign-gafgyt-mirai-unbalanced – A data sample set biased towards attack data that consists of about 10% benign, 45% Gafgyt and 45% Mirai data. Usually in the size of 2000 benign, 10000 Gafgyt and 10000 Mirai records

2.3 Feature selection algorithms

Since the main idea of the thesis was to research the use of feature selection in IoT botnet detection then several feature selection algorithms were tested during each part of the analysis. Following are the used algorithms for both unsupervised and supervised anomaly detection models.

2.3.1 Unsupervised anomaly detection

For the unsupervised anomaly detection models the feature selection algorithms used are also all unsupervised. This means that none of the methods look at what type each record is and treat them all equally.

2.3.1.1 Entropy

The first method used in selecting the best features is entropy. Entropy is a measure of uncertainty of a random outcome used here to measure the discriminatory power of different features. Entropy $E(v_i)$ of a random variable v_i is defined as [25]:

$$E(v_i) = -\sum_{j=1}^{k} p_j \log_2 p_j \quad (1)$$

Where p_j is the fraction of data points belonging to class j for attribute value v_i. After calculating the value-based entropies $E(v_i)$, the overall attribute-wise entropy E is defined as follows [25]:

$$E = \sum_{i=1}^{r} \frac{n_i E(v_i)}{n} \quad (2)$$

While normally entropy is used mainly for categorical values then with discretization it can be used for numeric variables as well [25]. In the current analysis, features resulting with higher entropy are selected.

2.3.1.2 Variance

Variance is a simple way of measuring how much does a feature vary within itself. Smaller variance scores mean that the data possesses similar values within a feature and therefore has less discriminatory power. This means that during feature selection the highest variance features should be selected [26].

2.3.1.3 Gini index

Similarly to entropy, Gini index is a way of measuring the discriminatory powers of certain features, that is mainly used for categorical values. It measures the impurity of how likely a randomly labelled element of a dataset would get labelled incorrectly [27]. Gini index $G(v_i)$ of a random variable v_i is defined as follows [25]:

$$G(v_i) = 1 - \sum_{j=1}^{k} {p_j}^2 \quad (3)$$

Where p_j is the fraction of data points belonging to class j for attribute value v_i. Like with entropy, after calculating the value-based Gini indices $G(v_i)$ the overall attribute-wise Gini index G is defined as follows [25]:

$$G = \sum_{i=1}^{r} \frac{n_i G(v_i)}{n} \quad (4)$$

In most cases Gini index and entropy achieve similar results with Gini index performing more efficiently, since unlike Entropy it does not have to compute the logarithm [25].

2.3.1.4 Hopkins statistic

Hopkins statistic is a way of measuring datasets clustering tendency by comparing sample data points distances to their nearest neighbors to the distances of a sample of synthetic randomly generated data points and their nearest neighbors [28]. For Hopkins statistic, let D represent the original dataset. For each feature of D, a sample R of r data records is selected from it. In addition to this, a synthetic data sample S of r randomly generated points is created from the data space of D. Hopkins statistic H is then defined as follows [25]:

$$H = \frac{\sum_{i=1}^{r} \beta_i}{\sum_{i=1}^{r} (\alpha_i + \beta_i)} \quad (5)$$

Where α_i are the distances of points in R to their nearest neighbors in D and β_i are the distances of points in the synthetic dataset S to their nearest neighbors in D.

The Hopkins statistic will end up in results between 0 and 1 with values around 0.5 meaning that the data is uniformly distributed. Other than that, values closer to 1 show a high cluster tendency and are therefore chosen during feature selection [25].

2.3.2 Supervised classification

While during supervised classification analysis labelled data is already available, meaning that also supervised feature selection methods can be used, then for the sake of comparison, all the previously described unsupervised feature selection methods will be used here as well. In addition to those, one supervised method, Fisher's score will be tested as well.

2.3.2.1 Fisher's score

Fisher's score is a supervised measure that can be used for feature selection by comparing the discriminatory powers of different attributes within data. The method selects features such that the differences between the same type data points would stay small while the differences between different class data points are as large as possible [29]. Fisher's score F is defined as follows [25]:

$$F = \frac{\sum_{j=1}^{k} p_j(\mu_j - \mu)^2}{\sum_{j=1}^{j} p_j \sigma_j^2} \quad (6)$$

Where μ_j is the mean-, σ_j is the standard deviation- and p_j is the fraction of class j data points and μ is the global mean of the data. The highest scoring features have the highest discriminatory power and are therefore chosen during feature selection [25].

2.4 Anomaly detection algorithms

The current work experiments with and analyses a variety of different unsupervised anomaly detection and supervised classification models to find answers to the main goals of the thesis. The goal of the methods is the same, for unsupervised models to correctly detect the anomalies in the data and for supervised methods to additionally correctly classify the type of a data point, between benign, Gafgyt and Mirai classes.

2.4.1 Unsupervised anomaly detection

The unsupervised anomaly detection methods work on only unlabeled data. This means that the training is done with benign records only and the anomaly detection methods then decide if an unseen data point belongs within the benign data or if it is an outlier [25]. Since evaluation of the methods uses both normal and attack data, then the validation of the anomaly detection models can be done similarly to supervised methods and is described more thoroughly in section 2.5.

2.4.1.1 One-Class SVM

One-class SVM, or one-class Support Vector Machine is an unsupervised learning model proposed by [30]. While the classic SVMs were originally supervised methods made for classification and regression analysis [25], then [30] presented a way of how the idea of SVM can be adapted to unsupervised learning as well by modifying a two-class SVM. SVMs work

by constructing an optimal hyperplane, that separates the distinct data points in a high-dimensional feature space [31]. With one-class SVM, the model is trained on one type of data only (in this case benign data) and the model uses that knowledge to learn the boundaries of the normal records and to separate outliers from normal data [30].

2.4.1.2 Isolation Forest

Second anomaly detection method that is used in the analysis is Isolation Forest (IF). Isolation forest is an unsupervised learning model proposed by [32], that is based on the idea of decision trees. It works by isolating the outliers by randomly selecting a feature from the data set and then splitting it from a random value between the maximum and minimum values. Doing this repeatedly will eventually isolate a data point from the others and create a tree structure where the leaves are the isolated data points. Since outliers are more likely to be separated from the rest in earlier partitions then shorter paths to the root in the generated trees are likely to represent the outliers [32].

2.4.1.3 Local Outlier Factor

The last unsupervised anomaly detection method used in the thesis, Local Outlier Factor (LOF), is a density-based detection algorithm. Proposed by [33], it works by comparing the densities of neighborhoods of each point in a data set based by the points *k* nearest neighbors. Calculating the LOF values for each point is a multi-step process that requires looking at the points distances to their nearest neighbors and the points average reachability distances from their neighbors. In the end, each point is left with a single value where higher LOF values indicating lower densities and therefore the probable outliers [25], [33].

2.4.2 Supervised classification

Unlike the previous anomaly detection methods, the supervised classification models require the training data to be labeled and for samples of data from each class to present in the training set. Instead of just detecting outliers, the supervised methods divide each data point to their respective class. In the current work, the methods are going to differentiate between benign, Gafgyt and Mirai records.

2.4.2.1 Decision Tree Classifier

The Decision Tree classifier is a supervised learning model that works by continuously splitting data into partitions based on learned decision rules, forming a tree shaped structure. During training of the model, the decisions on where the data should be split, also known as the split

criterions are made based on some of the feature variables, such that the class variables in each branch are as distinct as possible [34]. For example, having a split criterion that says that each record for which some specific feature has a value less than 1 goes to one branch and the other records for which the value is at least 1 move to another branch. Later, the classification of yet unseen records will be done based on the generated tree using top-down traversal from the root of the tree [25].

2.4.2.2 Random Forest

The Random Forest classifier creates several different discrete decision trees that all provide their own predictions. During training, random vectors are generated when building each decision tree. Later, during classification of yet unseen data records, each decision tree in the ensemble of trees creates their own prediction and the final decisions are made based on the most popular decisions of the many trees. This prevents individual errors showing up that a single decision tree might cause [25], [35].

2.4.2.3 K-Nearest Neighbors

The k-nearest neighbors (KNN) classification algorithm is one of the easiest to understand learning models that works on the principle that data points which are close to each other are also more similar [36]. KNN uses a *lazy learning* approach which means that during training an explicit model is not created based on the training data, but rather the training data is stored and is used during classification [37]. When classifying previously unseen data, the model looks at the labels of the nearest neighbors in the training data and bases the label for the new data point on the majority of its neighbors [36].

2.4.2.4 AdaBoost

AdaBoost is a supervised learning algorithm that uses a boosting approach. This means that the method uses a combination of many weak classifiers on iteratively modified versions of the initial data to train the model [25]. The model starts off by training weak classifiers with unmodified data. After evaluating the created model, all the data points get weights added to them, such that the incorrectly classified data points get assigned larger weights and the correctly classified points get assigned smaller weights than previously. A new weak classifier is then trained and evaluated with the new weighted data and new weights are assigned to the data points and the process is repeated. This process is iteratively run n times until the end results are then generated from a linear combination of each trained model [38].

2.4.2.5 MLP Classifier

While the current thesis does not center on neural network models, one such baseline method was still tested. The MLP classifier or the Multi-layer Perceptron is a supervised feedforward artificial neural network learning model. While normal perceptron models contain just a single input layer and an output layer, then the multi-layer model consists of additionally to the input and output layers of one or more non-linear hidden layers. The layers are connected to each other in such a way that the nodes in one layer feed into the nodes in the next layer [25].

2.4.2.6 Discarded methods (logistic regression, support vector machines)

In addition to the previously mentioned models, a few more were initially tested but later discarded from the final analysis because they did not on average achieve as good results as others. One of those models were the Support Vector Machines [25] that were briefly covered in section 2.4.1.1. While SVMs can be highly accurate depending on the objective then in the current work they performed worse than the other models. Similarly to SVMs, it was initially planned to incorporate Logistic Regression model [25] in the analysis as well, but it was abandoned quite early in the work because of low accuracy and performance.

2.5 Evaluation methods

Evaluation of the anomaly detection and classification methods results is done by calculating different performance indicators based on the models confusion matrices. Confusion matrix is a table which visualizes the correctly and incorrectly classified classes. On Table 1, a sample confusion matrix is shown.

Table 1. Two-class confusion matrix

	Predicted Benign	**Predicted Attack**
Actual Benign	True Positive (TP)	False Negative (FN)
Actual Attack	False Positive (FP)	True Negative (TN)

The true positive and true negative values show the amount of data points that got correctly classified as either benign or attack data, respectively. The false negative value shows the benign points that got incorrectly classified as attack and the false positive shows the attack points that got incorrectly classified as benign. Based on these results, different performance indicators can be calculated [39].

In the current work, four indicators are calculated for each model. The first one, accuracy, shows the ratio of correctly predicted data points to all data points. Accuracy is computed as following:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (7)$$

The second calculated parameter is precision, or the ratio of correctly predicted positive (benign) data points to all predicted benign data points. Precision is computed as following:

$$Precision = \frac{TP}{TP + FP} \quad (8)$$

The third parameter, recall, is the ratio of correctly predicted benign data points to all benign data points. Recall is computed as following:

$$Recall = \frac{TP}{TP + FN} \quad (9)$$

The last calculated indicator is f1-score, which finds the weighted average of precision and recall. F1-score is computed as following:

$$F1 - score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (10)$$

All these metrics can be easily found for the unsupervised anomaly detection part of the analysis where the models are only able to distinguish between benign and attack data [39]. However, during classification, the models divide each data record into one of the three available classes, benign, Gafgyt and Mirai. Because of this, the confusion matrix ends up having an extra row and column, which makes finding the TP, TN, FP, and FN not as straightforward. There are a few ways of how to deal with such multi-class confusion matrices and in the current work the following method is used. First, a separate confusion matrix is made for each class and separate performance indicators are calculated for them as well. This is done by assuming that for each class, the positive values are of that specific class and negative values are the data points that belong in either of the other classes. For example, a three-class confusion matrix for benign class is shown on Table 2. After the accuracies, precisions, recalls and f1-scores have been calculated for each class separately, simply an unweighted mean of these values is found as the final result [40].

Table 2. Three-class confusion matrix for benign class

	Predicted Benign	**Predicted Gafgyt**	**Predicted Mirai**
Actual Benign	TP	FN	FN
Actual Gafgyt	FP	TN	TN
Actual Mirai	FP	TN	TN

Lastly, in addition to the performance measures, the training time and testing time for each model is measured as well. The times are measured as the average times per instance, showing how much time did a model take per one training or testing data record.

2.6 Tools

The work was done in Python 3.8.5[1] language. Most of the experiments were done using the Python scikit-learn[2] library, which is an extensive machine learning and data analysis library. Graphs and plots were made using the matplotlib[3] library.

[1] https://www.python.org/

[2] https://scikit-learn.org/stable/

[3] https://matplotlib.org/2.0.2/index.html

3 Unsupervised anomaly detection analysis results

The first part of the analysis covers the unsupervised anomaly detection methods.

3.1 Analysis structure

The goal of the analysis is to answer the questions proposed previously: is it possible to reduce the number of anomaly detection models by training the models on multiple devices, how much does a prior knowledge of attack data affect the feature selection and the overall accuracy of anomaly detection models and which feature selection and anomaly detection methods achieve the highest accuracies. For this many different models were trained and tested and the results of said models are then compared.

3.1.1 Training and testing different models

The analysis focuses on comparing different models based on the proposed goals. To make the analysis easier and to prevent the possibility where some specific model that has not been tested yet is needed later, then initially many different models with all possible parameter combinations were trained. After this, a selected sample of these models are deeply analysed. The models chosen for the analysis are based on the specific objective of the test and is explained more thoroughly later. For each model, approximately 20000 benign data records are randomly chosen as the training dataset. Depending on the test, these records might all come from one device or they might be a mixed dataset from different devices. In addition to the training data, 30000 testing data records will be chosen as well. This testing data consists of equal amount of benign, Gafgyt and Mirai records and unlike the training data, these records all come from the same device.

For training and testing a model, a modified version of cross validation with a split size of 5 will be used. Usual cross validation cannot be used for the current tests, since that would require taking the training and testing data from the same initial dataset, such that after splitting the data one part will be used for testing and the rest will be used for training. However, for unsupervised model, the model needs to be trained with benign data only and then evaluated on both benign and attack data, making it impossible to choose the splits from a single dataset.

In addition to that, in many cases when training the model on a combined dataset and then testing it on a single device, the initial data comes from different datasets. The modified cross validation method will split both the training and testing dataset into n splits and then on each iteration selects n-1 splits from the training set as training data and 1 split from testing set as testing data. Based on the previously mentioned amount of data records randomly selected for each model, this means that on each iteration the model is trained on 16000 benign records and tested on 6000 mixed records.

3.1.2 Fixing values

Since the total number of different trained and tested models is large and each of those models have multiple specific parameters that must be taken into consideration, then it makes sense to look at each variable one at a time and use them to reduce the necessary varying data for later parts of the analysis.

For the first part of the test, some fixed values are taken for all, but one parameter and tests will be done on different values for this specific parameter. Doing this can show how much the end results differ when only changing one value. These fixed values can just be chosen as random but to try to prevent having to compare possibly highly inaccurate models, then some thought should be put into selecting them. There are a few different ways of how to choose the fixed values for the other parameters. First option would be to compare the average results for all models that used some specific value for a specific parameter. Based on the average results, one can fix the parameters to the values that were used in the models that resulted in highest average accuracies or f1-scores. For example, when selecting a feature selection algorithm, models with every other possible combination of other parameters (feature selection dataset, power of the feature set, anomaly detection algorithm, training dataset) will be tested and an average result of each of those models is computed. Those average results are then compared and the feature selection algorithm that was used in the most accurate models is fixed. This way, each possible value for a specific parameter would end up in the same number of tests, meaning that each value would have been tested equally. With this method however, there are still some possible caveats. For example, some specific values might result in highly accurate results when another parameter is also of some specific value but might perform poorly when the second parameter is of some other value. For example, using entropy as a mean of selecting features might result in great accuracy when using with an Isolation Forest anomaly detection algorithm but perform badly when combining it with Local Outlier Factor or One-Class SVM

instead. Another value however might cause mediocre results in all the tests, meaning that the average results of those tests end up higher than the previous value and therefore it will get selected more likely.

Another option to counter the previous issue, would be to compare all the results of all the tests and then just choose the value that was used in the most accurate model. This would also cause the selected values to not be 100% correct, since that might end up with results where one specific parameter resulted in a high accuracy results, but overall did not perform that well in other tests. Additionally, having investigated the resulting data already, usually the most accurate models resulted in very similar accuracies and f1-scores and therefore the differences might be within a probable error.

Third option would be to just research some previous papers written on similar topics and try selecting the fixed parameters based on that. However, since most of the research papers usually only cover a small sample of different parameters used here, then in the current thesis, the initial fixed values will be selected based on the average results.

3.1.3 Comparison of different models

After fixing all but one of the parameters, the results of the remaining models are then compared. For the analysis, initially all the accuracies and f1-scores of the different tested models are looked upon and compared with other models. In some cases, some additional analysis must be done to see how well a model did in correctly classifying benign or attack records by comparing the confusion matrices of few of the models. Additionally, for the tests where feature selection datasets or feature selection algorithms are compared, some extra attention must be paid to the actual features the currently selected methods chose. Lastly, since the goal of the analysis is to work with smaller and less powerful IoT devices and since it is necessary for these models to be able to detect attack data in real time applications, then some special focus should be given to the performance of the models as well. During the analysis, the testing times will be looked upon to see if the best accuracy models also perform quickly enough.

3.2 Training and testing dataset analysis

The analysis begins with one of the main questions proposed by the thesis, to find out, if it is possible to reduce the necessary computation power needed for anomaly detection by reducing

the total amount of trained models by creating a single model based on data received from multiple devices.

3.2.1 Choosing the fixed values

Comparing different device models requires taking fixed values for other possible parameters that can changed (feature selection algorithm, feature selection dataset, power of the feature set, anomaly detection algorithm) and comparing the results based on these values. To find out which fixed values to choose, the average f1-scores for every value of every parameter from thousands of tests will be compared and the values used in the overall most accurate models are selected.

3.2.1.1 Feature selection algorithm

From Figure 1, the average f1-scores of anomaly detection models corresponding to different feature selection algorithms can be seen.

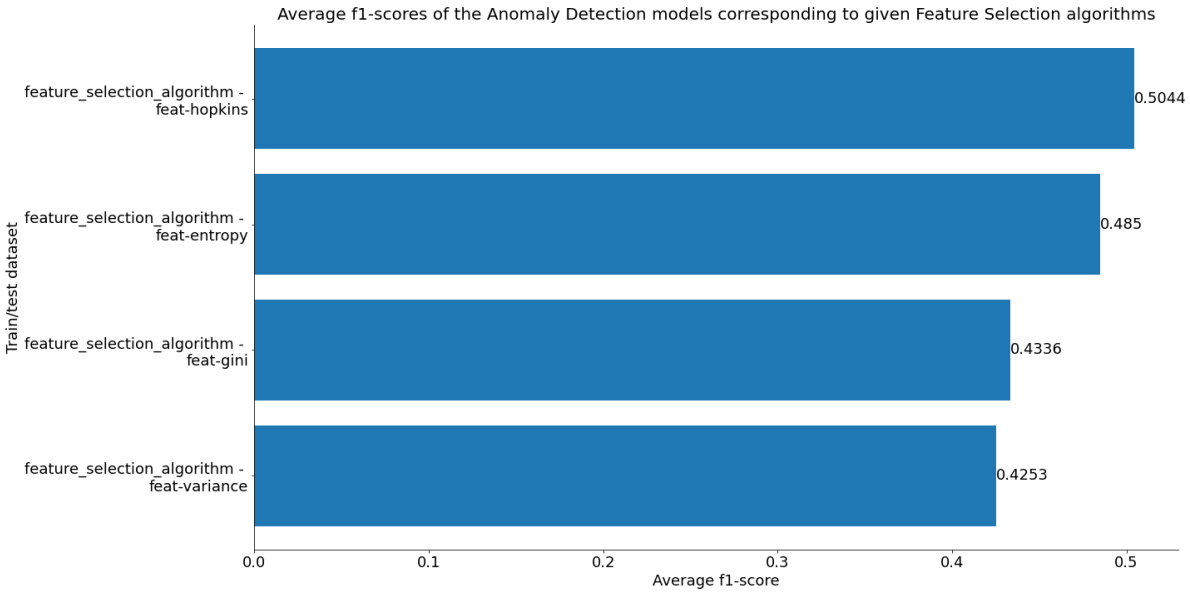

Figure 1. Average f1-scores of anomaly detection models corresponding to given feature selection algorithms

On Figure 1, models where features were selected by Hopkins statistic and Entropy both resulted in similar average f1-scores of about 48.5%-50%, with Hopkins statistic averaging just a little higher. Gini index and variance resulted in slightly lower f1-scores of about 43.3% and 42.5%, respectively. While the scores do seem rather low, this is expected since these results take every model into account, both the ones that gave optimal results as well as the ones that did not perform too well. Since the difference between the most accurate methods is not huge, then additionally looking at the average evaluation time it took for different tests to predict the

anomalies with different feature selection algorithms is necessary. The average testing times can be seen on Figure 2.

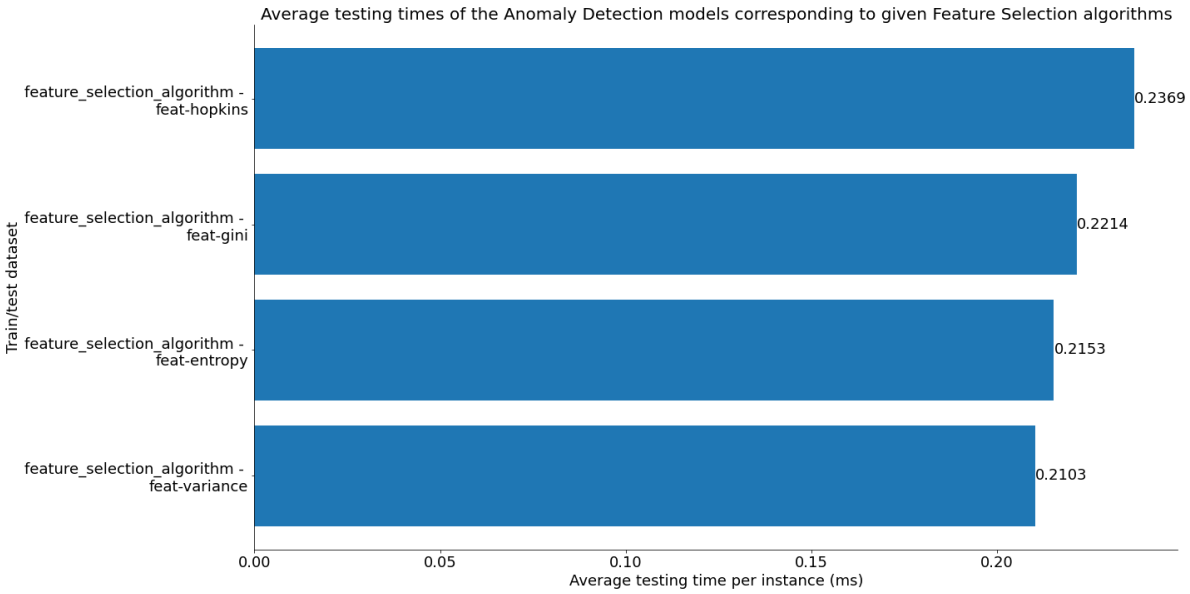

Figure 2. Average testing times of anomaly detection models corresponding to given feature selection algorithms

On Figure 2 on average, tests done with Hopkins statistic took a little longer than the other three methods with Gini index being the second slowest. This means that two algorithms stand out from the rest, Hopkins statistic with slightly higher accuracy and Entropy with a little faster evaluation time. Choosing a feature selection algorithm is highly based on what the current goals with the models are, whether it is important to find the fastest method that still averages in decent accuracy results or if it is more necessary to choose the most accurate model, even if it performs slower than the other methods. Based on these results, currently the higher accuracy method is chosen and therefore Hopkins statistic is fixed as the feature selection algorithm used in further analysis.

3.2.1.2 Feature selection dataset

On Figure 3, the average f1-scores of anomaly detection models corresponding to different feature selection datasets can be seen.

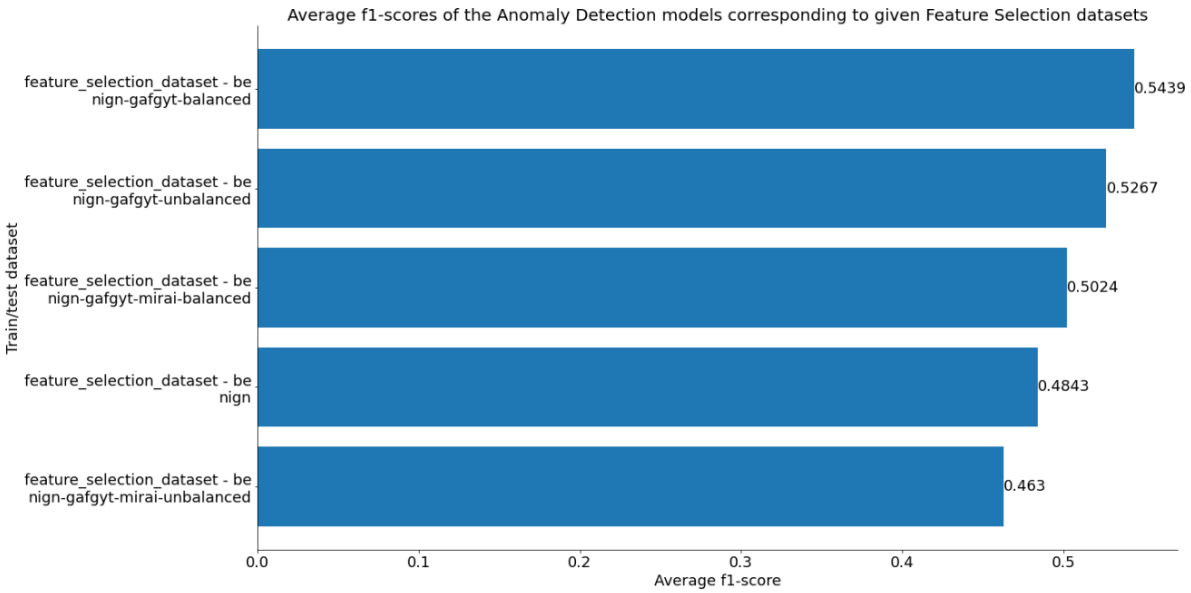

Figure 3. Average f1-scores of anomaly detection models corresponding to given feature selection datasets

From Figure 3, both first two datasets, that the features were chosen from, that provided the best scoring anomaly detection models consisted of just benign and Gafgyt data with no Mirai data in it. On average the results of all the models ended up looking quite similar to the previous with each dataset averaging about 2% higher f1-score than the next one on the list. While 2% difference can be large, especially when dealing with critical data, then since the current results only show the averages of hundreds of tests and the f1-scores for each of the models are not very high, then here the difference might not be as important. While benign-gafgyt-balanced data sample averaged in higher f1-score than the rest, then its unbalanced counterpart averaged in the highest accuracy overall. However, the benign-gafgyt-unbalanced dataset is very much skewed towards Gafgyt data, with it having only about 10% of benign data in it. Since usually in real world, outside of experimental networks benign data is much more common with most of the devices never having encountered attack data at all before, then again, looking at the average testing times and seeing if these correlate with the most accurate datasets is necessary. The average testing times can be seen on Figure 4.

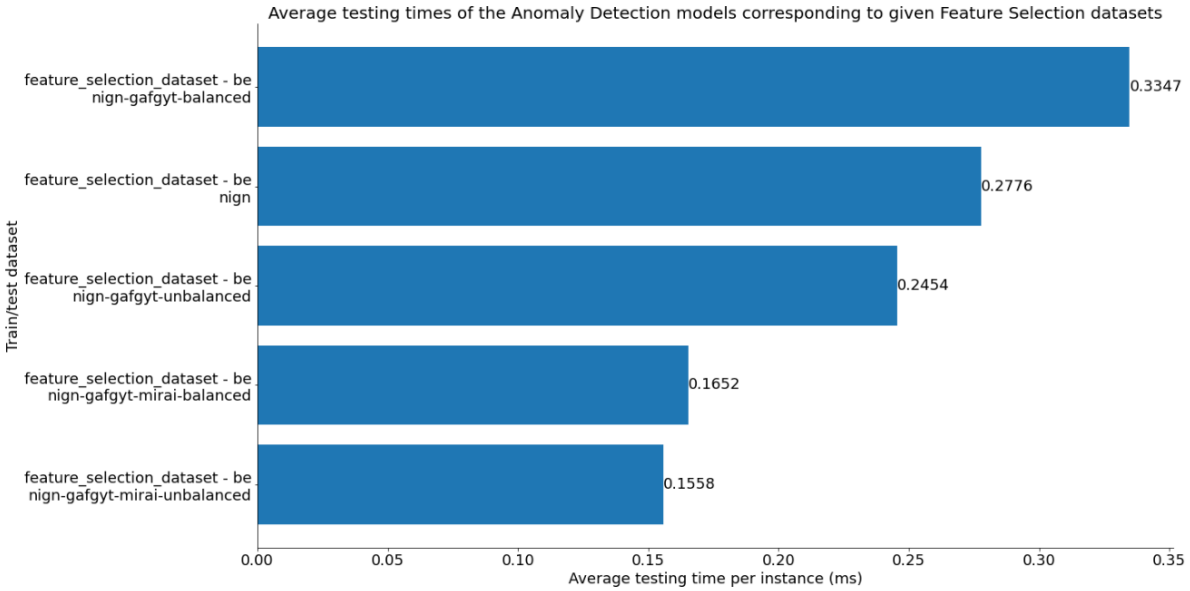

Figure 4. Average testing times of anomaly detection models corresponding to given feature selection datasets

Interestingly, the average model evaluation times tell a completely different story. From Figure 4 it can be seen that both models that used all three types of data for feature selection, provided with considerably quicker results than the other datasets. Just like with feature selection algorithm, a choice between more accurate results or faster results must be made. Because the unbalanced datasets are biased towards attack data, then currently these will not be looked at and the fixed value will be chosen from the more balanced samples. Since the time difference between the two balanced samples is quite high, then the current choice for the dataset fixed value will be made from the evaluation times. Benign-gafgyt-mirai-balanced data based classification model was able to predict the anomalies about twice as fast as its benign-gafgyt-balanced counterpart, meaning that it will be used during feature selection.

3.2.1.3 Power of the feature set

On Figure 5, the average f1-scores of anomaly detection models corresponding to different feature set powers can be seen.

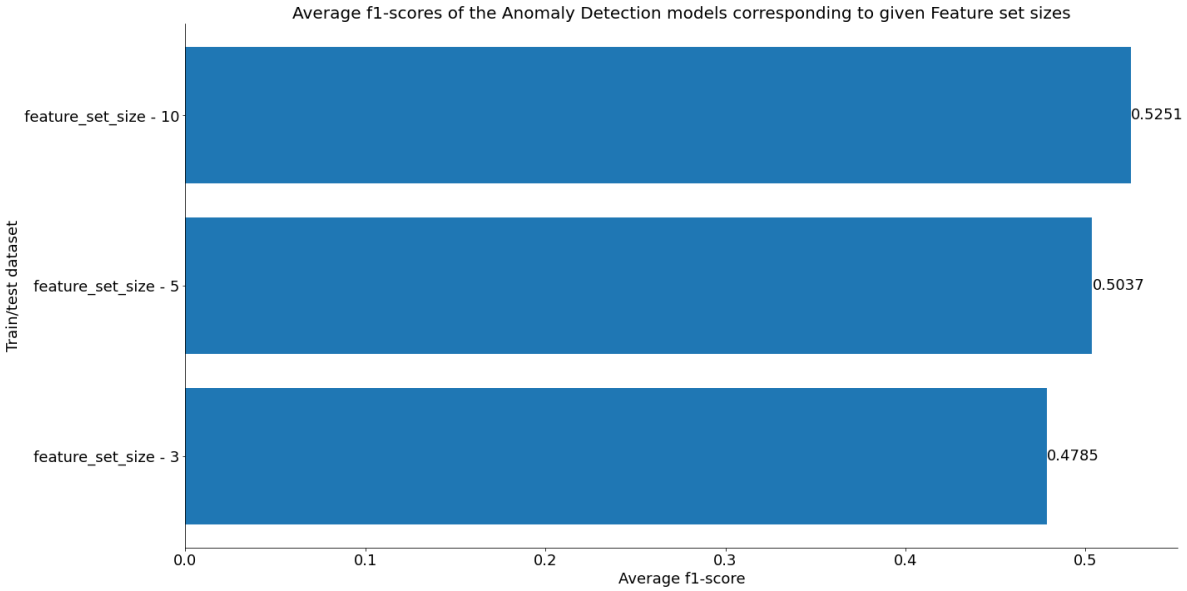

Figure 5. Average f1-scores of anomaly detection models corresponding to given feature set powers

From Figure 5, on average models based on all three feature set sizes of 3, 5 and 10 resulted in quite close f1-scores with the bigger sizes giving higher results than the rest, meaning that again looking at the average training times will hopefully give a definite option. The average testing times can be seen on Figure 6.

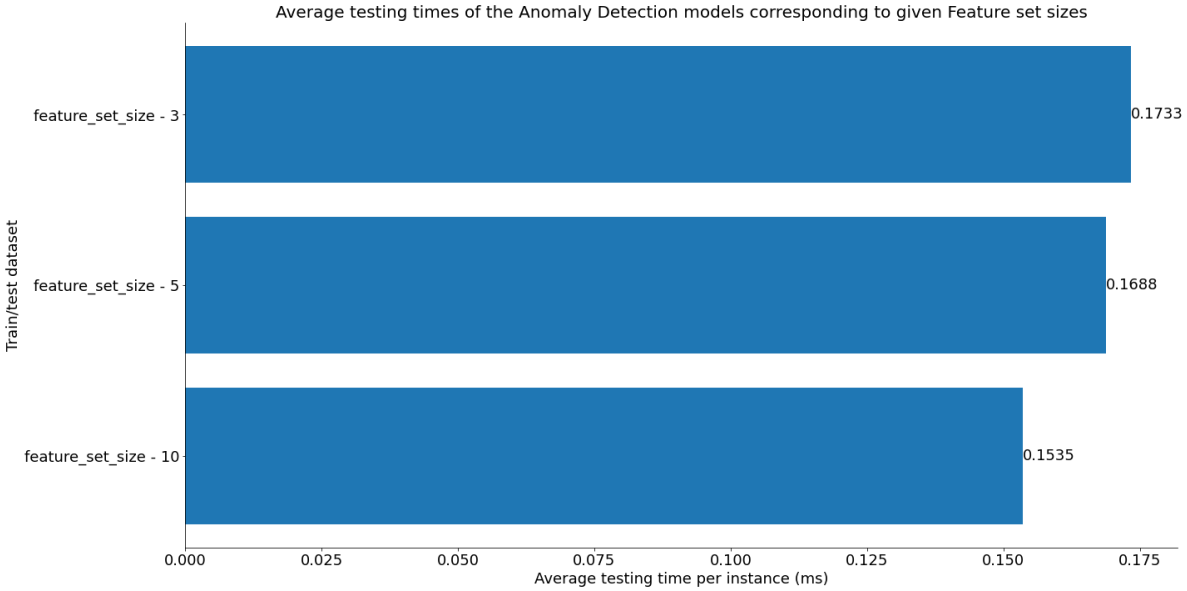

Figure 6. Average testing times of anomaly detection models corresponding to given feature set powers

From the graph the performance results correlate with the f1-scores, with the most accurate feature size also averaging in the lowest testing time. Because of this, the number of selected features for the initial tests will be fixed as 10.

3.2.1.4 Anomaly detection algorithm

On Figure 7, the average f1-scores of anomaly detection models corresponding to different anomaly detection algorithms can be seen.

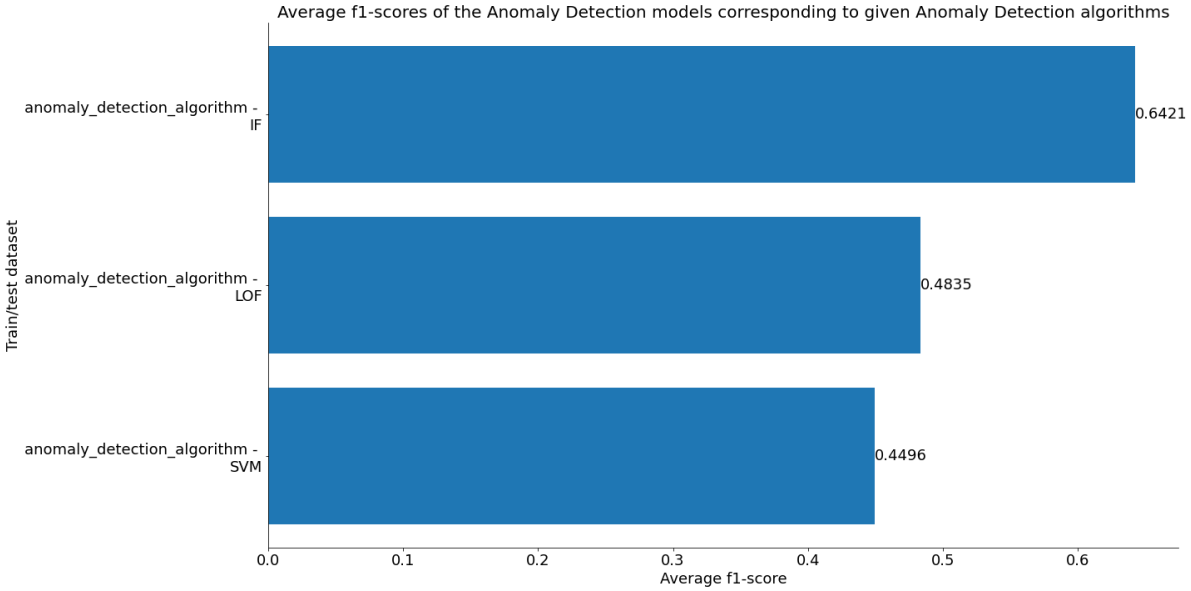

Figure 7. Average f1-scores of anomaly detection models corresponding to given anomaly detection algorithms

Unlike the previous three parameters, on Figure 7, a clear distinction between different values can be seen with Isolation Forest performing on average noticeably better than the other two anomaly detection algorithms and SVM resulting in the least accurate results. Although a clear distinction between the algorithms can be seen, looking at the evaluation times would still help to see if the choice would also make sense on the performance part. The average testing times can be seen on Figure 8.

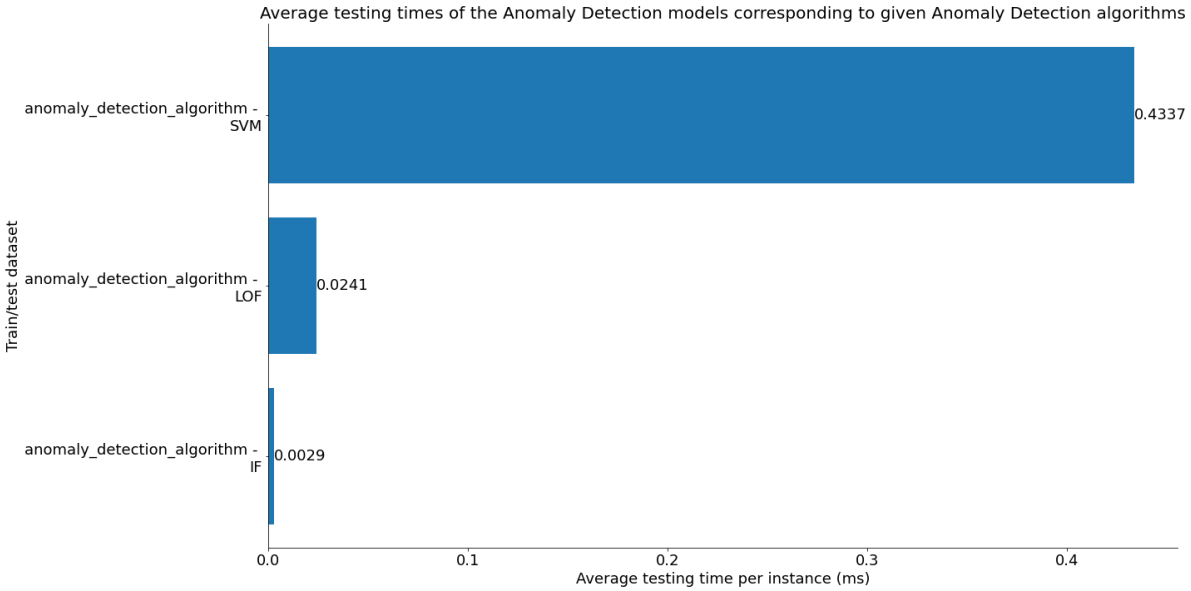

Figure 8. Average testing times of anomaly detection models corresponding to given anomaly detection algorithms

From Figure 7 and Figure 8, the models that used the most accurate algorithm also took the least amount of time. Based on the results Isolation Forest can be fixed as the anomaly detection model.

After analysing the average results across all possible tests, the fixed values that the comparison of different training and testing datasets will be carried out on has been chosen. The fixed values are shown in Table 3.

Table 3. Selected fixed values for training and testing dataset analysis

Feature selection algorithm	Hopkins statistic
Feature selection dataset	Benign-gafgyt-mirai-balanced
Power of the feature set	10
Anomaly detection algorithm	Isolation Forest

3.2.2 Comparison of anomaly detection models trained and tested on different devices

Now that other parameters have fixed values, analysing, and comparing different training and testing dataset pairs can begin. As mentioned before, currently there are 54 total pairs of training and testing datasets available. The training will be carried out on all devices together, five types of different cameras together, two types of different SimpleHome security cameras together, two types of Provision security cameras together, two types of doorbells together and separate models will be created for every single device as well. All the models, except the ones

trained on just one device, will then be evaluated on all the devices separately. This means that for the following analysis the following aspects should be considered. First, and most importantly, it would be necessary to only choose a subsample of those 54 trained models so not every model has to be compared. For this, first thing that can be done is to exclude all the models that were trained on a combined dataset of multiple devices and evaluated on a device that was not present in the initial training set, since those models would most likely not result in the best scores. For example, using a model that was trained on cameras and then evaluated against Ecobee thermostat can be excluded. After removing all those models, 29 out of 54 are still left. From the leftover models some additional restrictions will be made to reduce the number of models needed in the analysis and to make the whole analysis more uniform for every device used. Currently out of the nine devices, four are available in four different models training sets, three devices are available in three different models and two devices are available in just two models. By removing the models that used the two combined security cameras datasets in training, the leftover models become more uniform, making the analysis easier. While before all four security cameras were present in four models, then after removing some of them, the 25 remaining models are divided as seen on Figure 9. Now, out of the nine devices, seven are used in three different models training set and two are used in just two different models.

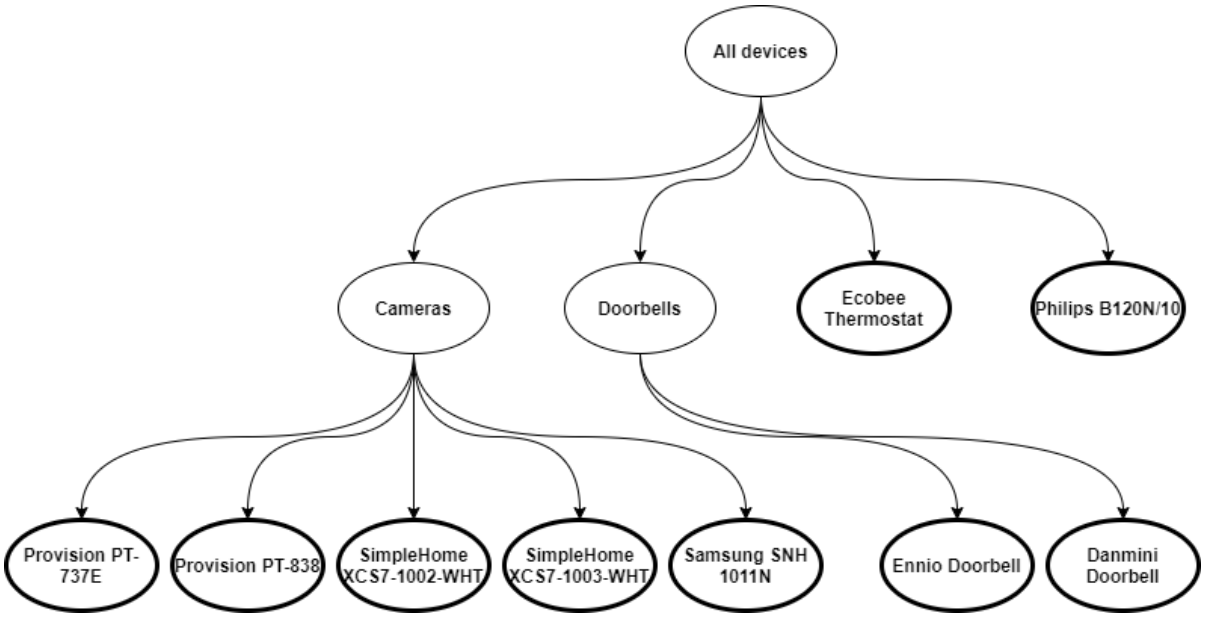

Figure 9. Structure of the different models training sets

3.2.2.1 Anomaly detection models results

After the number of models were reduced to 25, the results of all said models can then be compared. On Table 4 the accuracies, f1-scores, and testing times per instance for each model are shown. The results are also illustrated on Figure 10.

Table 4. Results for Isolation Forest anomaly detection models trained and tested on different devices

Testing Device	**Category**	**Model trained on all-devices**			**Model trained on different device categories (cameras, doorbells)**			**Model trained on single device**		
		Accuracy	**F1-score**	**Testing time (ms)**	**Accuracy**	**F1-score**	**Testing time (ms)**	**Accuracy**	**F1-score**	**Testing time (ms)**
Provision PT-737E	Cameras	0.81207	0.75147	0.00281	0.78713	0.70603	0.00278	0.78140	0.70079	0.00253
Provision PT-838	Cameras	0.81127	0.74199	0.00299	0.79170	0.70264	0.00302	0.78193	0.69838	0.00300
SimpleHome XCS7-1002-WHT	Cameras	0.82267	0.77622	0.00336	0.80030	0.74811	0.00277	0.62627	0.62337	0.00360
SimpleHome XCS7-1003-WHT	Cameras	0.82447	0.78692	0.00333	0.79323	0.74838	0.00289	0.86463	0.82760	0.00301
Samsung SNH 1011N[4]	Cameras	0.58835	0.59761	0.00468	0.67715	0.69174	0.00774	0.88800	0.88792	0.00426
Ennio doorbell[4]	Doorbells	0.82090	0.84390	0.00662	0.74585	0.76270	0.00652	0.42410	0.59483	0.00297
Danmini doorbell	Doorbells	0.85593	0.82039	0.00283	0.60227	0.63270	0.00303	0.74670	0.62121	0.00257
Ecobee thermostat	-	0.76027	0.68304	0.00285	-	-	-	0.88873	0.84566	0.00250
Philips B120N/10	-	0.82690	0.78793	0.00269	-	-	-	0.71800	0.68770	0.00263
Average		0.81090	0.77323	0.00308	0.70243	0.67561	0.00298	0.74664	0.72083	0.00301

[4] No Mirai data available

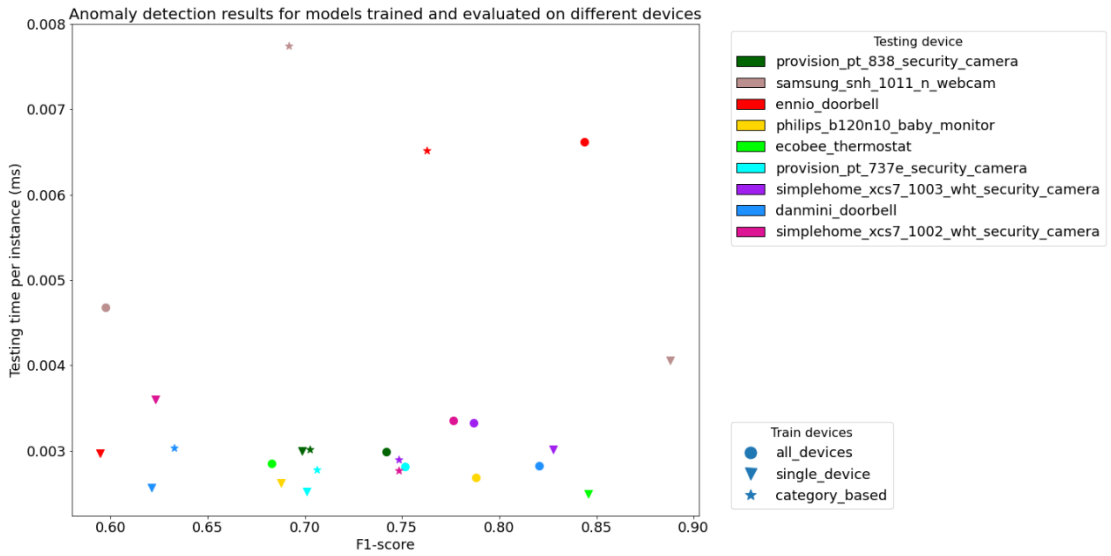

Figure 10. f1-scores and average testing times of different anomaly detection models

Interestingly, when looking at the results, for most of the devices, the models that were trained on multiple devices managed to result in higher accuracies and f1-scores than the models trained on just the single device that the model was also evaluated against. These results are unexpected, since the more complex training sets of all devices and category-based training data have less data records from each specific device compared to the single device model. Additionally, the more complex models use training data from other devices, not related to the testing device that might cause the models to incorrectly classify some records. From the nine devices, for only three did the single device model predict anomalies with a higher accuracy than the all-devices model. These devices are the SimpleHome XCS7-1003-WHT security camera, the Samsung SNH 1011N web camera and the Ecobee thermostat, which resulted in f1-scores of 82.8%, 88.8% and 84.6% for the single device model compared to the results of the all-devices model of 78.7%, 59.8% and 68.3% respectively. However, while comparing the all-devices and the single device models did end up with different results based on the device the model was evaluated against then the category-based model did almost always perform worse than the all-devices model. The only exception for this was Samsung SNH 1011N, where the all-devices model achieved an f1-score of 59.8% compared to cameras model f1-score of 76.3%. Since the category-based models are the middle ground between more complex models that would require less maintenance and the single device models that should perform better in theory then in the current tests for the most part they did not get the advantage in either side.

Looking at the confusion matrices for some of the models that got opposite results, where for one the single device model achieved the highest accuracy compared to another where the all-devices model got the highest accuracy can hopefully tell how such results were achieved. For this, the confusion matrices for both types of models are presented for both SimpleHome security cameras in Table 5 - Table 8.

Table 5. Confusion matrix for anomaly detection model trained on all devices and evaluated on SimpleHome XCS7-1002-WHT data

	Predicted benign	**Predicted attack**
Actual benign	9229	771
Actual attack	4549	15451

Table 6. Confusion matrix for anomaly detection model trained on SimpleHome XCS7-1002-WHT data and evaluated on SimpleHome XCS7-1002-WHT data

	Predicted benign	**Predicted attack**
Actual benign	9276	724
Actual attack	10486	9514

Table 7. Confusion matrix for anomaly detection model trained on all devices and evaluated on SimpleHome XCS7-1003-WHT data

	Predicted benign	**Predicted attack**
Actual benign	9725	275
Actual attack	4991	15009

Table 8. Confusion matrix for anomaly detection model trained on SimpleHome XCS7-1003-WHT data and evaluated on SimpleHome XCS7-1003-WHT data

	Predicted benign	**Predicted attack**
Actual benign	9116	884
Actual attack	3177	16823

From the confusion matrices both all-devices models got similar results. While the SimpleHome XCS7-1003-WHT model was able to correctly classify slightly more benign datapoints of about 97.3% (9725 out of 10000) compared to the other device's 92.3% (9229 out of 10000) then the SimpleHome XCS7-1002-WHT model predicted more attack records correctly of about 77.3% (15451 out of 20000) compared to the other device's 75% (15009 out of 20000). However, when looking at the results of the models that were trained on one device something curious appears. While the model trained on

SimpleHome XCS7-1002-WHT data classified a similar number of benign records correctly to the all-devices model then now the model did much worse in predicting attack data. The model classified more than half of the attack records as benign, causing an accuracy of only about 47.6% (9514 out of 20000). For the SimpleHome XCS7-1003-WHT the results seemed to be the exact opposite. The single device model did worse in correctly classifying normal records at about 91.2% (9116 out of 10000) but managed to achieve higher results in predicting attack data at about 84.1% (16823 out of 20000).

3.2.2.2 Anomaly detection models performance

In addition to the accuracies the models achieved, special attention should be given to the testing times as well. Since for IoT devices performance is much more important than when just testing the methods on a more powerful PC then it is necessary to see if the models resulting in higher accuracies do not suffer from the performance perspective. For most of the models the testing times do not differ from each other by a large margin. The testing times are mostly around 0.003 millisecond per instance with some models performing slightly faster and others slightly slower. However, there are still a few outliers that take longer than the rest. The overall highest accuracy model, the one that was both trained and tested on the Samsung web camera took around 0.004 milliseconds to predict the attack records with other Samsung web camera models taking even longer up to 0.0077 ms. Another device that resulted in longer times was the Ennio doorbell, which all-devices and category-based models both took around 0.006 ms. Interestingly, the slower models were all based on the devices that did not have any Mirai data available.

3.2.3 Summary of the models

After extensive tests of comparing different anomaly detection models to find out if it is possible to efficiently reduce the total amount of trained models by creating a single model based on data received from multiple devices, it still cannot be answered with absolute confidence that works for every situation. While for most devices creating combined training datasets did seem to produce satisfactory results, and even increase the accuracy of the created models, then this was not the case for every single model that got tested. For few of the devices, the single device model did prove to be more accurate than the rest. On the performance part most of the models did achieve similar times and therefore all of them could be possibly used in real time anomaly detection. However, the current analysis only compared the results of a smaller subset of models where every

other parameter, other than the training and testing devices was given a fixed value. Changing those fixed values might completely change the overall outcome of the models.

3.3 Feature selection dataset analysis

The second main goal of the current thesis was to see if and how much does having prior knowledge of different types of attack data affect the results of feature selection and overall performance of the unsupervised and supervised anomaly detection models. Because of this, a series of tests will be carried out on similar data where the only parameter that will be changed is the data sample the best features are chosen from. Just like with the previous tests where different devices were compared, every other parameter will be fixed.

3.3.1 Choosing the fixed values

Since most of the parameters were already chosen by the previous training data analysis, then mostly the values will be used here as well, such as the anomaly detection algorithm and the feature selection algorithm. Additionally, the devices from where the training and testing datasets were taken from need to now be fixed as well. For this the same average function used when fixing the other parameters could be used again or the chosen datasets can be based on the previous results of the training data analysis. Although the previous testing results did not cover every single training and testing device pair, then the fixed devices will be still chosen from there by selecting the training and testing device pair that resulted in higher accuracy. While the SimpleHome XCS7-1003-WHT model was not the overall highest accuracy model, then since the more accurate models were either present in less datasets or did not have any Mirai attack data available, then the single-device model for the SimpleHome security camera was selected. The chosen fixed values are shown in Table 9.

Table 9. Selected fixed values for feature selection dataset analysis

Training device	SimpleHome XCS7-1003-WHT
Testing device	SimpleHome XCS7-1003-WHT
Feature selection algorithm	Hopkins statistic
Power of the feature set	10
Anomaly detection algorithm	Isolation Forest

3.3.2 Comparison of anomaly detection models with different feature selection datasets

With the other parameters fixed, analysing how different amounts and types of benign and attack data can affect the outcome of feature selection can start. Unlike the previous part of the analysis where two parameters, training device and testing device, did not have a fixed value set, then this time only one of the parameters can still vary. This means that there is no need to choose a subsample of the already largely sampled parameters and all the values can be tested.

For analysing the different data that will be used for feature selection, first a sample of data will be taken from the training dataset. This sample will then be split into parts: only benign data; benign and Gafgyt data in balanced form; benign and Gafgyt data in unbalanced form where most of the data comprises of Gafgyt; benign, Gafgyt and Mirai data in balanced form and benign, Gafgyt and Mirai data in unbalanced form where most of the data comprises of Gafgyt and Mirai data. After this, the same series of feature selection and anomaly detection methods will be carried out on for all. It should also be noted that all the models will be evaluated on the same mixed dataset.

3.3.2.1 Benign data

For the first part the features were chosen strictly from benign data. This is usually most common in real life applications where a device has yet to previously encounter any attack data. After feature selection the following features from Table 10 were chosen.

Table 10. Selected features for benign data

HH_L3_covariance
HH_jit_L5_mean
HH_jit_L5_variance
HH_jit_L3_mean
HH_jit_L3_variance
HH_jit_L1_variance
HpHp_L5_covariance
HpHp_L5_pcc
HpHp_L3_covariance
HpHp_L3_pcc

From the list of selected features by Hopkins statistic for benign data, most of the choices belonged to either the channel jitter (HH_jit) or socket (HpHp) categories with the exception of one channel (HH) based feature. All of the features also came from smaller time-frames of mostly 100ms and 500ms with one feature from 1.5s time-frame.

After the features had been selected, Isolation Forest model was applied to detect the anomalies. The anomaly detection model results and the confusion matrix are seen in Table 11 and Table 12, respectively.

Table 11. Results for anomaly detection model with 10 features selected by Hopkins statistic for benign data

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time (ms)**	**Training time (ms)**
0.7928	0.66389	0.91768	0.75735	0.004626	0.002963

Table 12. Confusion matrix for anomaly detection model with 10 features selected by Hopkins statistic for benign data

	Predicted Normal	**Predicted Attack**
Actual Normal	9177	823
Actual Attack	5393	14607

The Isolation Forest model was able to predict the anomalies from a dataset where the best features were chosen by Hopkins statistic from benign data only with about 79.3% accuracy and 75.7% f1-score and with an average testing time of approximately 0.0046 milliseconds per instance. From the confusion matrix the model was able to correctly predict a decent portion of benign cases at about 91.8% (9177 out of 10000) but failed to perform at predicting attack records, with it only classifying about 73% of these cases correctly.

3.3.2.2 Benign and Gafgyt data balanced

Here, the features were chosen from a mixed data of benign and Gafgyt records with equal amounts of both types of data. After feature selection the following features from Table 13 were chosen.

Table 13. Selected features for balanced benign and Gafgyt data

HH_L5_covariance
HH_L1_covariance

HH_jit_L5_mean
HH_jit_L5_variance
HH_jit_L3_mean
HH_jit_L1_mean
HH_jit_L0.1_mean
HH_jit_L0.01_mean
HpHp_L3_magnitude
HpHp_L0.01_radius

From the list of selected features by Hopkins statistic for benign and Gafgyt data in balanced form, some differences with the previous only benign data can be seen. This time the most discriminating features came mostly from channel jitter category (HH_jit), with two features from channel and two features from socket category. While most of the selected features again came from smaller time-frames, then this time also three larger time-frame features scored higher Hopkins statistic results than the rest.

The anomaly detection model results and the confusion matrix are seen in Table 14 and Table 15, respectively.

Table 14. Results for anomaly detection model with 10 features selected by Hopkins statistic for benign and Gafgyt balanced data

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time**	**Training time**
0.79433	0.66625	0.90427	0.75747	0.003892	0.004941

Table 15. Confusion matrix for anomaly detection model with 10 features selected by Hopkins statistic for benign and Gafgyt balanced data

	Predicted Normal	**Predicted Attack**
Actual Normal	9043	957
Actual Attack	5213	14787

The Isolation Forest model based on data where best features were chosen from a balanced dataset of benign and Gafgyt records achieved quite similar results to the first part of the analysis. The model scored an accuracy of about 79.4% and f1-score of about 75.7% with an average testing time of approximately 0.0039 milliseconds per instance. The confusion matrix does not differ from the first part by a large margin either, with the

current model being able to correctly predict about 90.4% (9043 out of 10000) of benign data and 73.9% (14787 out of 20000) of attack cases.

3.3.2.3 Benign and Gafgyt data unbalanced

Here, the features were chosen from a mixed data of benign and Gafgyt records with more Gafgyt records available. After feature selection the following features from Table 16 were chosen.

Table 16. Selected features for unbalanced benign and Gafgyt data

HH_L5_mean
HH_L5_radius
HH_L3_mean
HH_jit_L5_mean
HH_jit_L5_variance
HH_jit_L3_mean
HH_jit_L3_variance
HH_jit_L0.01_mean
HpHp_L5_std
HpHp_L3_magnitude

When biasing the feature selection sample more towards Gafgyt data, the selected features seemingly started moving away from socket category and more towards channel jitter and channel. This time five of the ten most discriminating features belonged to channel jitter, three features belonged to channel and just two belonged to socket categories. The time-frames were still mainly on the smaller side with just one feature obtained from the largest time-frame available.

The Isolation Forest model results and the confusion matrix are seen on Table 17 and Table 18, respectively.

Table 17. Results for anomaly detection model with 10 features selected by Hopkins statistic for benign and Gafgyt unbalanced data

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time**	**Training time**
0.81173	0.65830	0.90610	0.76242	0.003759	0.002488

Table 18. Confusion matrix for anomaly detection model with 10 features selected by Hopkins statistic for benign and Gafgyt unbalanced data

	Predicted Normal	**Predicted Attack**
Actual Normal	9061	939
Actual Attack	4709	15291

With the anomaly detection model where the best features were chosen from an unbalanced set of benign and Gafgyt cases the results improved slightly. The accuracy and f1-score of said model rose to approximately 81.2% and 76.2% respectively with an average testing time of about 0.0038 milliseconds per instance. While the amount of correctly predicted benign cases stayed close to the previous models at around 90.6% (9061 out of 10000) then the current model was able to correctly classify a slightly larger amount of attack records at 76.5% (15291 out of 20000).

3.3.2.4 Benign, Gafgyt and Mirai data balanced

Here, the features were chosen from a mixed data of benign, Gafgyt and Mirai records with equal amounts of all types of data. After feature selection the following features from Table 19 were chosen.

Table 19. Selected features for balanced benign, Gafgyt and Mirai data

HH_L5_radius
HH_L3_radius
HH_L3_covariance
HH_L1_covariance
HH_jit_L5_mean
HH_jit_L5_variance
HH_jit_L3_mean
HH_jit_L0.01_mean
HpHp_L3_mean
HpHp_L3_magnitude

With the addition of Mirai data, the selected features started skewing towards channel category. Most of the features now came from channel and channel jitter categories with again just two features belonging to socket. The time-frames however did not change a lot, with most of the features still obtained from shorter times.

For anomaly detection, Isolation Forest was able to predict the records with the following results and confusion matrix seen on Table 20 and Table 21.

Table 20. Results for anomaly detection model with 10 features selected by Hopkins statistic for benign, Gafgyt and Mirai balanced data

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time**	**Training time**
0.86463	0.77332	0.91164	0.82760	0.003013	0.001973

Table 21. Confusion matrix for anomaly detection model with 10 features selected by Hopkins statistic for benign, Gafgyt and Mirai balanced data

	Predicted Normal	**Predicted Attack**
Actual Normal	9116	884
Actual Attack	3177	16823

With the addition of Mirai data to the feature selection sample set the results of the Isolation Forest model did go up a step. The new model now achieved an accuracy of about 86.5% and an f1-score of about 82.8% with also a faster time of 0.003 milliseconds per instance. Looking at the confusion matrix the correctly predicted normal samples stayed similar to the previous cases with the model being able to classify 91.2% (9116 out of 10000) of benign records correctly. However, the greatest enhancement came from predicting the attack records, with the model correctly classifying about 84.1% (16823 out of 20000) of the attack cases which is almost an 8% improvement from the currently second-best model.

3.3.2.5 Benign, Gafgyt and Mirai data unbalanced

Here, the features were chosen from a mixed data of benign, Gafgyt and Mirai records with more Gafgyt and Mirai records available. After feature selection the following features from Table 22 were chosen.

Table 22. Selected features for unbalanced benign, Gafgyt and Mirai data

HH_L5_mean
HH_L5_radius
HH_L3_mean
HH_L1_radius
HH_jit_L5_mean
HH_jit_L3_mean

HpHp_L5_mean
HpHp_L5_std
HpHp_L3_magnitude
HpHp_L0.01_radius

With the last, biased towards attack data, data sample most of the discriminating features did not belong to channel jitter category anymore, but were mostly of the channel and socket type. However, just like with last tests, the features still mostly belonged to the smaller time-frames.

The Isolation Forest model results and the confusion matrix are seen on Table 23 and Table 24, respectively.

Table 23. Results for anomaly detection model with 10 features selected by Hopkins statistic for benign, Gafgyt and Mirai unbalanced data

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time**	**Training time**
0.94763	0.99724	0.84530	0.91493	0.003615	0.002658

Table 24. Confusion matrix for anomaly detection model with 10 features selected by Hopkins statistic for benign, Gafgyt and Mirai unbalanced data

	Predicted Normal	**Predicted Attack**
Actual Normal	8453	1547
Actual Attack	24	19976

After adding Mirai data to the feature selection data sample set and biasing that data more towards attack records, the anomaly detection model was able to produce even greater results. This time the model improved even further and was able to predict the anomalies with an accuracy of about 94.8% and an f1-score of about 91.5%. The testing time went back up slightly, at around 0.0036 milliseconds per instance. The confusion matrix shows that this time the model did classify a larger number of normal records incorrectly. While every previous model was able to predict over 90% of benign cases correctly then the last model did so only at around 84.5% (8453 out of 10000) accuracy. However, the model was able to achieve great results at classifying attack data with it correctly predicting almost 99.9% (19976 out of 20000) of the cases. Only 24 attack records got incorrectly classified as normal.

3.3.3 Summary of the models

In Table 25, the distribution of the best 10 features selected by different feature selection datasets according to their categories can be seen.

Table 25. Distribution of the best 10 features selected by different feature selection datasets

Category	Benign	Benign, Gafgyt balanced	Benign, Gafgyt unbalanced	Benign, Gafgyt, Mirai balanced	Benign, Gafgyt, Mirai unbalanced
Source IP					
Source MAC-IP					
Channel	1	2	3	4	4
Channel jitter	5	6	5	4	2
Socket	4	2	2	2	4

Looking at the selected features from all the tests, it can be observed that the feature selection does in fact differ from whether a model has only benign data available compared to also having some prior knowledge of attack data. When with just benign data most of the selected features belonged to either channel jitter or socket categories, then with adding more attack data and removing benign data the most discriminating features started curving more towards channel category and away from channel jitter.

The overall results of the five different models can be seen on Table 26. From these results, a few interesting points can be seen. Comparing the baseline model, where features were selected on benign data only with the two models where Gafgyt data was added to the feature selection as well the results did not differ by a large margin. For benign and benign, Gafgyt balanced models, the accuracies and f1-scores stayed approximately the same. When biasing the data more towards Gafgyt data the results did improve slightly but the change was not very large. The testing times did also get slightly quicker with the biggest difference seen between the model where features were selected on benign data only and the rest of the models. However, after introducing Mirai data to the feature selection dataset as well, the results did start improving. Model, where features were selected on a balanced dataset of all three types of data managed to predict anomalies with over 5% higher accuracy and over 6% higher f1-score than the previous best model. Biasing that data towards the two types of attack data, so only about 10% of the records in the feature selection dataset were from benign data caused the model to perform even

better. The accuracy and f1-score for the final model had risen from the baseline model for over 15%, the accuracy from 79.3% to 94.8% and the f1-score from 75.7% to 91.5%. However, while the final model did achieve high accuracy, the number of false negative predictions, where model predicted normal data to be of attack data did grow almost twice as high. This means that depending on the purpose of the model, the final model might not be the best choice if correctly labelling benign data is as or more important than correctly detecting attack records. Other than that, the model where feature selection was done on the balanced dataset of all three types of data managed to do better on both sides. The model did end up with a little more false positive records than the baseline benign model did, but the difference was not as big as with the other models. However, this model also improved in correctly detecting attack records.

Table 26. Results for anomaly detection model with 10 features selected by Hopkins statistic from different data sample sets

Feature selection data sample set	**Accuracy**	**F1-score**	**Testing time per instance (ms)**
Benign	0.79280	0.75735	0.00463
Benign and Gafgyt balanced	0.79433	0.75747	0.00389
Benign and Gafgyt unbalanced	0.81173	0.76242	0.00376
Benign, Gafgyt and Mirai balanced	0.86463	0.82760	0.00301
Benign, Gafgyt and Mirai unbalanced	0.94763	0.91493	0.00362

3.4 Feature selection algorithm analysis

Although the main goals of the thesis were to analyse how much does the knowledge of attack data affect the results of feature selection and if it is possible to create a common model for multiple IoT devices to reduce the maintenance costs, then a smaller goal of testing and comparing other parameters of unsupervised and supervised anomaly detection methods was proposed as well. Because of this, a series of similar tests will be carried out for selecting a feature selection algorithm, feature set size and an anomaly detection algorithm.

3.4.1 Choosing the fixed values

Starting with selecting the optimal feature selection algorithm, first all other parameters need to be fixed once again. Just like before, since most of the parameters were already chosen in the previous parts of the analysis, then the same will be used here as well. The training and testing devices are chosen from the first part of the analysis results, the feature selection dataset is chosen from the second part of the analysis results and the power of the feature set and the anomaly detection algorithms will be chosen from the average results. The chosen fixed values are shown in Table 27.

Table 27. Selected fixed values for feature selection algorithm analysis

Training device	SimpleHome XCS7-1003-WHT
Testing device	SimpleHome XCS7-1003-WHT
Feature selection dataset	Benign, Gafgyt and Mirai data balanced
Power of the feature set	10
Anomaly detection algorithm	Isolation Forest

3.4.2 Comparison of anomaly detection models with different feature selection algorithms

With the other parameters fixed, analysing how different feature selection algorithms attached to the same datasets can affect the outcome of feature selection can start. For the analysis, first a sample of data is taken from the training dataset. After that, 10 features are chosen based on the results of the four different feature selection algorithms: highest entropy, highest variance, highest Gini index and highest Hopkins statistic score. Every other feature that did not get chosen is discarded. Then different anomaly detection algorithms are applied to the reduced datasets and the results are compared.

3.4.2.1 Entropy

Here, the 10 features that resulted in the highest entropies are chosen. These chosen features can be seen in Table 28.

Table 28. Features chosen by highest entropy

HH_L1_weight
HH_L1_std
HH_L1_radius
HH_L0.1_weight

HH_L0.1_std
HH_L0.1_radius
HH_L0.01_std
HH_L0.01_radius
HH_jit_L1_weight
HH_jit_L1_variance

From the list of selected features by highest entropy, most of the choices belonged to the channel (HH) category apart from two channel jitter (HH_jit) based features. All of the features also came from larger time-frames of 1.5s to 1min.

For anomaly detection, Isolation Forest was able to predict the records with the following results seen on Table 29.

Table 29. Anomaly detection results for entropy feature selection algorithm

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time**	**Training time**
0.67503	0.50714	0.90990	0.65123	0.003719	0.002637

The Isolation Forest model was able to predict the anomalies from a dataset where the best features were chosen by Entropy only with about 67.5% accuracy and 65.1% f1-score and with an average testing time of approximately 0.0037 milliseconds per instance.

3.4.2.2 Variance

Here, the 10 features that resulted in the highest variances are chosen. These chosen features can be seen in Table 30.

Table 30. Features chosen by highest variance

HH_jit_L5_mean
HH_jit_L5_variance
HH_jit_L3_mean
HH_jit_L3_variance
HH_jit_L1_mean
HH_jit_L1_variance
HH_jit_L0.1_mean
HH_jit_L0.1_variance
HH_jit_L0.01_mean
HH_jit_L0.01_variance

When selecting features based on the highest variance only channel jitter category features got chosen with an equal number of features from every time-frame.

For anomaly detection, Isolation Forest was able to predict the records with the following results seen on Table 31.

Table 31. Anomaly detection results for variance feature selection algorithm

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time**	**Training time**
0.84550	0.71550	0.93340	0.80603	0.003242	0.002342

Compared to entropy, the model where features were chosen by the highest variance managed to perform with much higher accuracy of about 85.5% and f1-score of about 80.6%. On average the model was able to predict the anomalies at 0.003 millisecond per instance.

3.4.2.3 Gini index

Here, the 10 features that resulted in the highest Gini indices are chosen. These chosen features can be seen in Table 32.

Table 32. Features chosen by highest Gini index

MI_dir_L0.1_mean
MI_dir_L0.01_mean
HH_L3_mean
HH_L0.1_magnitude
HH_L0.01_mean
HH_L0.01_magnitude
HpHp_L5_mean
HpHp_L5_magnitude
HpHp_L3_mean
HpHp_L3_magnitude

With Gini index, the features were mostly selected from channel and socket categories with two features also from Source MAC-IP. The features all came from different time-frames with most of the source MAC-IP and channel features from larger time-frames and socket features from smaller time-frames.

For anomaly detection, Isolation Forest was able to predict the records with the following results seen on Table 33.

Table 33. Anomaly detection results for Gini index feature selection algorithm

Accuracy	**Precision**	**Recall**	**F1 score**	**Testing time**	**Training time**
0.91093	0.92952	0.82216	0.86683	0.00344	0.002313

With features selected by Gini index, the Isolation Forest model performed even better, with an accuracy of 91.1% and f1-score of 86.7%. Testing took a similar amount of time to the previous models at around 0.0034 milliseconds per instance.

3.4.2.4 Hopkins statistic

Here, the 10 features that resulted in the highest Hopkins statistic are chosen. These chosen features can be seen in Table 34.

Table 34. Features chosen by Hopkins statistic

HH_L5_radius
HH_L3_radius
HH_L3_covariance
HH_L1_covariance
HH_jit_L5_mean
HH_jit_L5_variance
HH_jit_L3_mean
HH_jit_L0.01_mean
HpHp_L3_mean
HpHp_L3_magnitude

For Hopkins statistic the results are the same as in the previous feature selection dataset analysis. Most of the features are chosen from smaller time-frames and from channel and channel jitter categories with two features also coming from the socket category.

For anomaly detection, Isolation Forest was able to predict the records with the following results seen on Table 35.

Table 35. Anomaly detection results for Hopkins statistic feature selection algorithm

Accuracy	Precision	Recall	F1 score	Testing time	Training time
0.86463	0.77332	0.91164	0.82760	0.003013	0.001973

With Hopkins statistic, the model achieved less accurate results than with Gini index but beat both Entropy and Variance. The model resulted in accuracy of about 86.5% and f1-score of 82.8% with the average testing time of 0.003 milliseconds per instance.

3.4.3 Summary of the models

In Table 36, the distribution of the best 10 features selected by different feature selection algorithms according to their categories can be seen.

Table 36. Distribution of the best 10 features selected by different feature selection algorithms

Category	Entropy	Variance	Gini index	Hopkins statistic
Source IP				
Source MAC-IP			2	
Channel	8		4	4
Channel jitter	2	10		4
Socket			4	2

From the table a clear distinction between different models can be seen. Both Entropy and Hopkins statistic seemed to select features mainly from channel and channel jitter categories with entropy favouring channel category. With variance every feature was chosen from the channel jitter category. This is also supported by the study [6], where a similar conclusion was found. The only method that picked any other category other than channel, channel jitter or socket was Gini index, which selected two features from the source MAC-IP category. The model where this method was used also provided with the best overall results. None of the methods selected any source IP features.

The overall results of the four different models can be seen on Table 37. From there, model where feature selection was done with Gini index performed with an accuracy of almost 5% better than the next best model. The difference between the accuracies of the next two models, where features were selected by Hopkins statistic and variance respectively ended up smaller, with approximately 2% between them. By a large margin,

the worst result was achieved by entropy, that predicted anomalies with over 17% lesser accuracy than the variance-based model. Comparing the results with the selected features the best performing model was the only one which selected two host-based MAC-IP features, possibly showing that the given feature category possesses higher discriminatory power than the rest. The next two best methods, both chose more features from the channel jitter category and the worst performing model chose features mainly from the channel category which indicates a lower discriminatory power. These findings do correlate with the study [6], where it was also found out that host-based and channel jitter categories had higher discriminatory power than the rest.

Table 37. Results for anomaly detection model with 10 features selected by different feature selection methods

Feature selection method	**Accuracy**	**F1-score**	**Testing time per instance (ms)**
Entropy	0.67503	0.65123	0.00372
Variance	0.84550	0.80603	0.00324
Gini index	0.91093	0.86683	0.00344
Hopkins statistic	0.86463	0.82760	0.00301

3.5 Power of the feature set and anomaly detection algorithm analysis

For the last part of the initial analysis, both two remaining parameters, power of the feature set and anomaly detection algorithm will be looked at together.

3.5.1 Choosing the fixed values

Since both, the feature set size and anomaly detection algorithm have a total of 3 possible values in the current tests, and since often the results of different anomaly detection methods can vary by the amount of features the input data has, then all 9 different combinations of the two parameters will be analysed. Every other parameter will receive their fixed value from the previous tests. The chosen fixed values are shown in Table 38.

Table 38. Selected fixed values for feature set powers and anomaly detection algorithm analysis

Training device	SimpleHome XCS7-1003-WHT
Testing device	SimpleHome XCS7-1003-WHT
Feature selection dataset	Benign, Gafgyt and Mirai balanced
Feature selection algorithm	Gini index

3.5.2 Comparison of anomaly detection models with different feature set powers and anomaly detection algorithms

With the other parameters fixed, the analysis of how different amounts of features and different unsupervised anomaly detection algorithms can affect the outcome of anomaly detection model can begin. For the analysis, first a sample of data is taken from the training dataset. After that, 3, 5 or 10 features are chosen based on the results of the selected feature selection algorithm. The current anomaly detection algorithm is then trained and evaluated on the reduced datasets and the results are compared. In Table 39, the results of each power of the feature set and anomaly detection method combination can be seen.

Table 39. Anomaly detection results for different feature set powers and anomaly detection algorithms

Feature set power	Anomaly detection algorithm	Accuracy	Precision	Recall	F1-score	Testing time per instance	Training time per instance
3	IF	0.90543	0.93146	0.80294	0.85582	0.00353	0.00241
5	IF	0.93853	0.99975	0.81584	0.89841	0.00367	0.00241
10	IF	0.91093	0.92952	0.82216	0.83383	0.00344	0.00231
3	SVM	0.83423	1	0.50271	0.66899	0.62274	0.95802
5	SVM	0.83437	1	0.50311	0.66933	0.63472	0.96814
10	SVM	0.83480	1	0.50442	0.67052	0.58520	0.86618
3	LOF	0.69833	0.52629	0.95281	0.67803	0.01232	0.02399
5	LOF	0.69727	0.52545	0.94971	0.67655	0.01826	0.04460
10	LOF	0.69717	0.52537	0.94904	0.67630	0.03059	0.06398

Looking at the table, none of the three anomaly detection model results differed by a great margin when changing the amount of features the models were trained and evaluated on. For both One-Class SVM and Local Outlier Factor the accuracies and f1-scores differed by less than 0.5% across all three feature set sizes. The biggest differences appeared with Isolation Forest, where the 5-feature model achieved an accuracy of about 93.9% and f1-score of 89.8% compared to lower accuracies of 90.5% and 91.1% and f1-scores of 85.6% and 83.4% for feature set powers of 3 and 10, respectively. Comparing the overall results of the anomaly detection methods, Isolation Forest achieved the best accuracies and f1-scores across the table. The models were able to correctly classify 93%-99% of the attack records, while averaging at 80%-82% accuracy in correctly predicting normal records. When looking at the other two methods, it seems that SVM managed achieve about 14%

better accuracies than LOF. However, the f1-scores are very close for both of the methods with LOF resulting in little higher scores. From precision and recall values it appears that while SVM did a great job in correctly classifying attack records (every attack record got correctly classified in all three models) then it also had trouble in detecting normal cases. All three SVM models classified approximately half of the normal records as attack data. For LOF however, the case was exactly the opposite. While LOF classified about 95% of the normal records correctly, it only managed to predict about 52.5% of the attack data.

On the testing time side, Isolation Forest models performed the fastest with LOF averaging 10 times slower times and SVM about 200 times slower times. This means that for the selection of anomaly detection method, the Isolation Forest stands out. Not only did it achieve the overall best results in predicting anomalies, but it also did that the quickest. For the power of the feature set, the choice however is not as clear, since all three sizes got similar results with the 5-feature model resulting in better f1-score than its 3 or 10 feature counterparts.

4 Supervised classification analysis results

The second part of the analysis covers the supervised classification methods.

4.1 Analysis structure

The goal of the analysis and most of the structure used in the analysis are the same as it was for the anomaly detection part. Just like previously, the analysis looks for answers to the goals proposed by the thesis by creating many classification models and then compares them based on some specific parameters. All the other parameters will get assigned some fixed values by using the same average results method described before.

4.1.1 Training and testing different models

One of the main changes between the unsupervised anomaly detection and the supervised classification analysis is the way of how training and evaluating the models is conducted. Since the tested models are all supervised, then the training is done with mixed data of benign and both types of attack data.

Most of the training and testing uses a similar modified cross validation method as described in paragraph 3.1.1. This is because for a lot of the models, the training set and testing set contains a combination of different data. For example, a model might be trained on a combined dataset of data from multiple devices but then evaluated on a single device only.

For models that are trained on a single device and tested on the same device, a more accurate bootstrap method can be used instead. A bootstrap method is a statistical method used for training and evaluating given models. While with cross validation the input data was randomly split into *n* splits such that each split contained different records than the rest and that each record was only present once in one split at the same time, then with bootstrap initially a random data sample is taken with replacement. Selecting a sample with replacement means that after randomly selecting a data record from the dataset it will not be removed from the initial data and is available for further selections as well.

The sample set that gets chosen after a specific number of iterations is called a bootstrap sample and most of the time the size of the bootstrap sample is the same as the original data set. Since the samples are chosen with replacement, then the bootstrap sample can contain some data records more than once and some others are not represented in the bootstrap sample at all. On average, a bootstrap sample with the size of the initial dataset contains approximately 63.2% of the original data records [41]. This generated bootstrap sample is then used during training of the classification models. All the records that did not get selected are the out-of-bag samples and are used for then evaluating the trained models. This process then gets repeated as many times as necessary and the mean of the results from all the models is found as the final result [41], [42]. In the current work, a bootstrap sample size of the original data size is selected, and the process is done for 20 iterations.

4.2 Training and testing dataset analysis

The supervised classification analysis is set up similarly to the unsupervised models with starting off with one of the main goals of comparing models trained on multiple devices and models trained on a single device.

4.2.1 Choosing the fixed values

To analyse different models, parameters other than the training and testing dataset must be set to a fixed value. Selecting these fixed values will be based on the average f1-scores and the average testing times of the models the values were used in.

4.2.1.1 Feature selection algorithm

On Figure 11, the average f1-scores of supervised classification models where different feature selection algorithms were used can be seen.

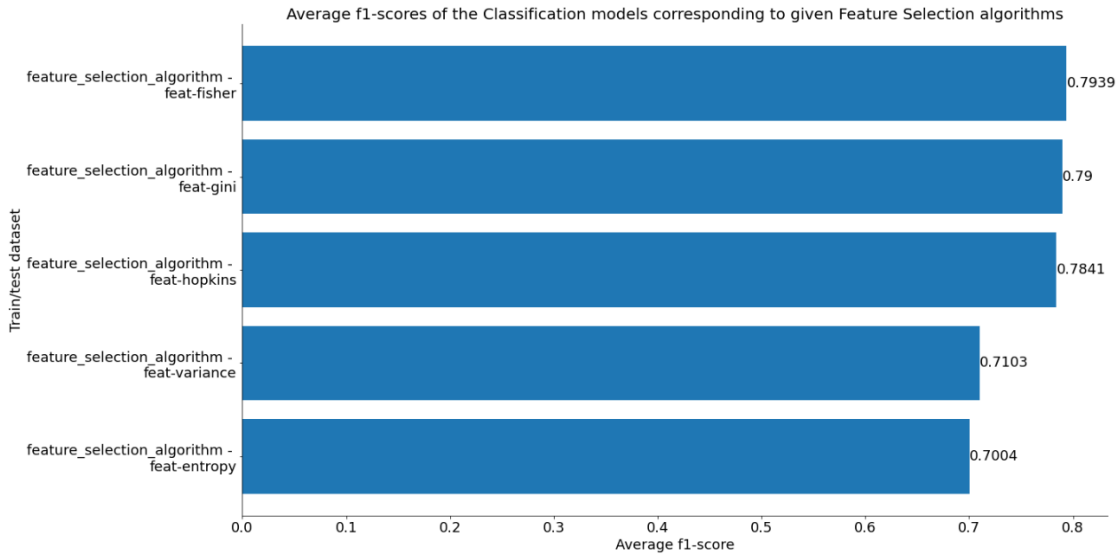

Figure 11. Average f1-scores of classification models corresponding to given feature selection algorithms

From Figure 11, models where features were chosen by Hopkins statistic, Gini index and Fisher's score, all resulted on average in similar f1-scores of around 79%, meaning that two of the four tested unsupervised feature selection algorithms were able to select equally discriminating features as the supervised counterpart. The average testing times for each model can be seen on Figure 12.

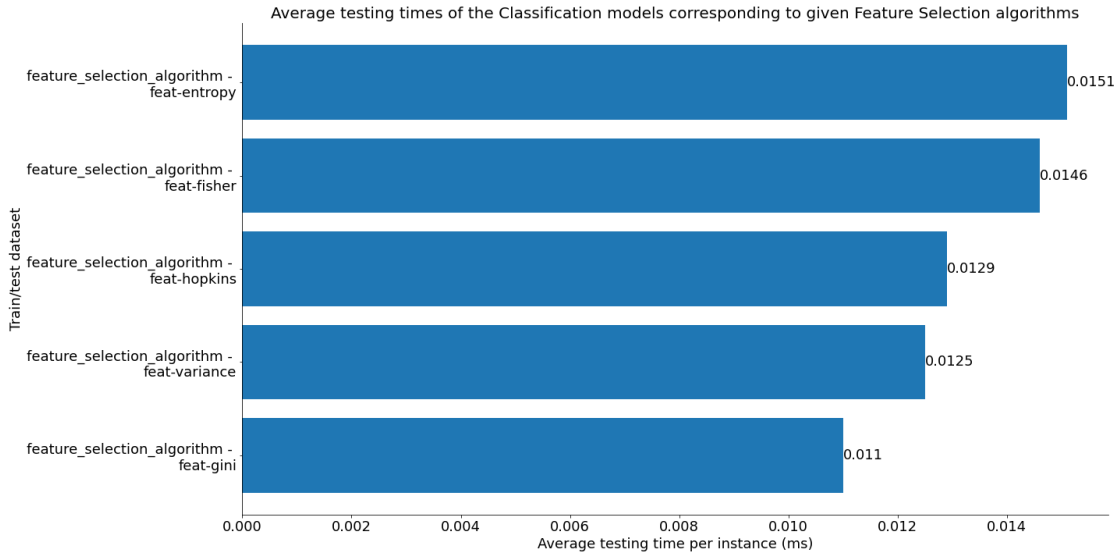

Figure 12. Average testing times of classification models corresponding to given feature selection algorithms

Looking at the testing times, all models achieved fast results with very small time differences between models. From the three most accurate models, the model where features were selected by Gini index achieved just slightly faster results than the rest. However, since the time difference is small and since none of the three best scoring

feature selection algorithms resulted in much better classification models than the rest then the selection for the fixed value can be made freely. Based on these findings, the only supervised feature selection method, Fisher's score, is chosen and used in further analysis.

4.2.1.2 Feature selection dataset

On Figure 13, the average f1-scores of classification models corresponding to different feature selection datasets can be seen.

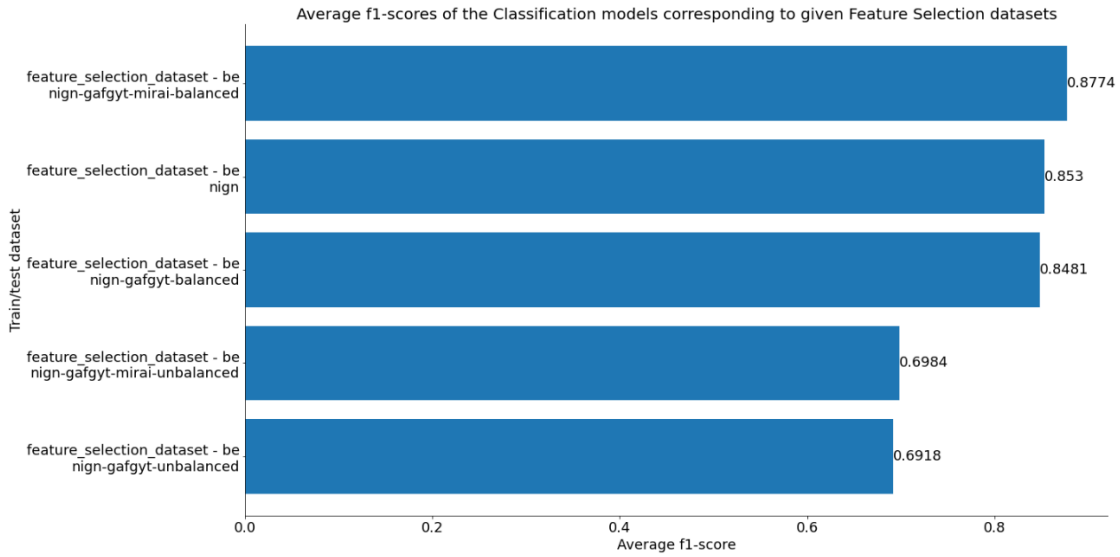

Figure 13. Average f1-scores of classification models corresponding to given feature selection datasets

From Figure 13, three feature selection datasets on average resulted in higher scoring classification models than the rest. The three methods resulted in f1-scores of 85-88% with both models based on the unbalanced datasets scoring noticeably lower results. From the best models, the model where features were chosen from all three data types resulted in on average 2% higher f1-score than the next best model. While the benign-gafgyt-mirai-balanced dataset did result in the most accurate models, then looking at the testing times will show if that selection would make sense at the performance part as well. The average testing times can be seen on Figure 14.

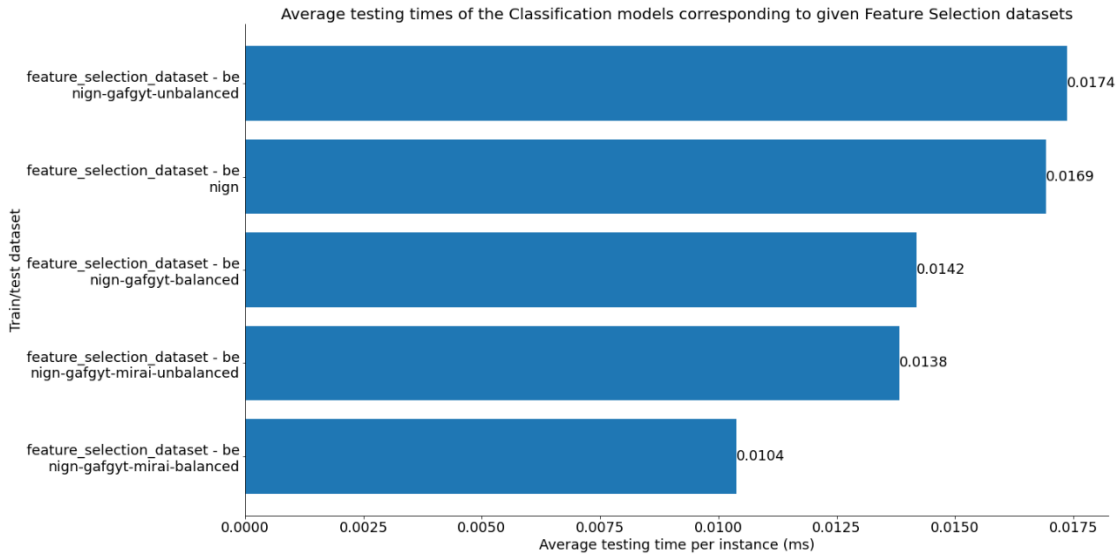

Figure 14. Average testing times of classification models corresponding to given feature selection datasets

From the graph it can be quickly seen that the fastest models were also the ones that resulted in the highest average f1-scores. Because of this, the benign-gafgyt-mirai-balanced dataset can be fixed as the feature selection dataset for further analysis.

4.2.1.3 Power of the feature set

On Figure 15, the average f1-scores of anomaly detection models corresponding to different feature set powers can be seen.

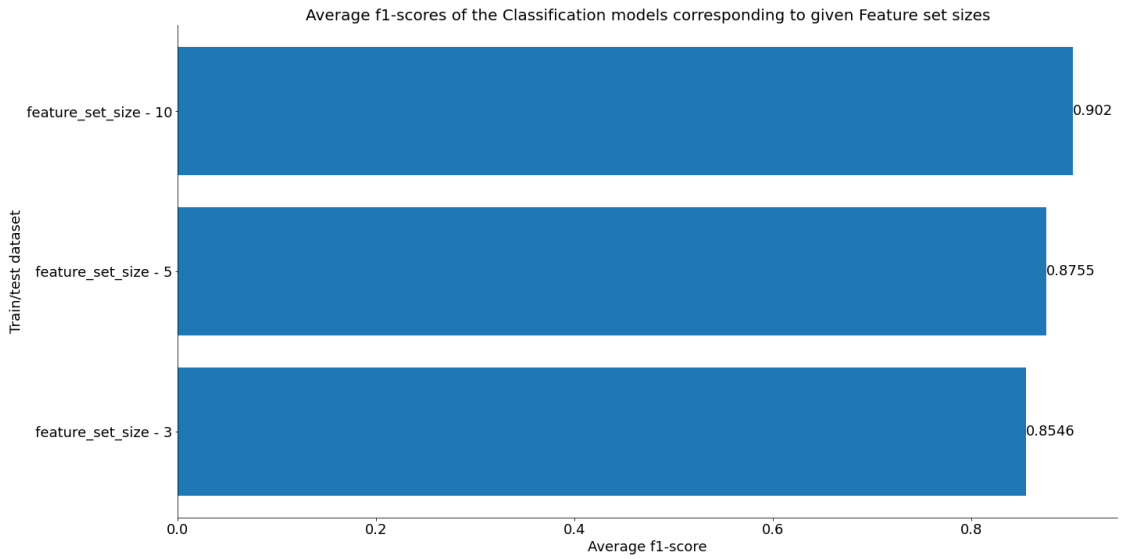

Figure 15. Average f1-scores of classification models corresponding to given feature set powers

From the figure, as expected, the models which were based on more features managed to result in more accurate results than the others, where models with 10 features resulted in

an average f1-score of about 90.2%, 5 features of about 87.5% and 3 features of about 85.5%. The average testing times for these models can be seen on Figure 16.

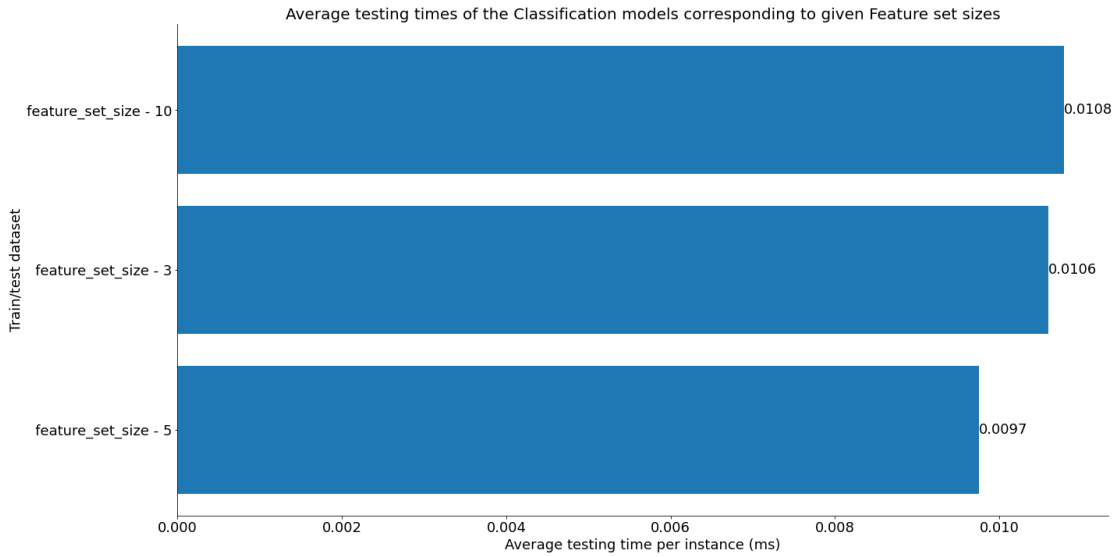

Figure 16. Average testing times of classification models corresponding to given feature set powers

All the models with different feature set powers were able to classify records in very similar times with no apparent correlation between the number of features and the testing time. While the highest scoring classification models did take the longest amount of time during testing, the time differences are not huge and because of this, the power of selected features can be fixed as 10.

4.2.1.4 Classification algorithm

On Figure 17, the average f1-scores of classification models corresponding to different classification algorithms can be seen.

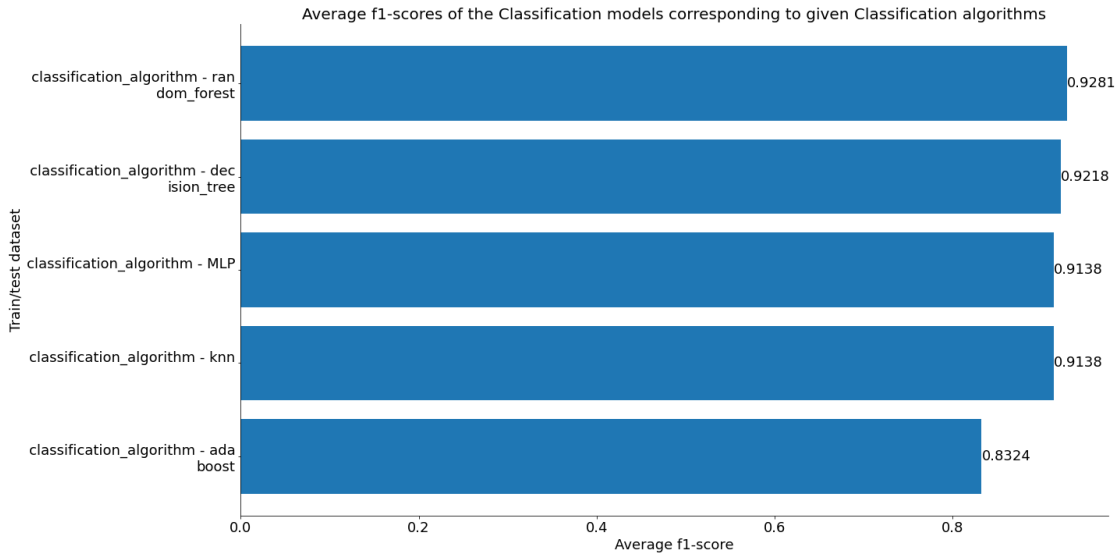

Figure 17. Average f1-scores of classification models corresponding to given classification algorithms

From Figure 17, four out of the five tested supervised classification methods resulted on average in equally accurate models with f1-scores of about 91.4%-92.8%. Only one of the algorithms, Adaboost, averaged approximately 8% lower results than the next. Since not a clear distinction between four of the models can be made, then looking at the performance of the models is necessary. The average testing times can be seen on

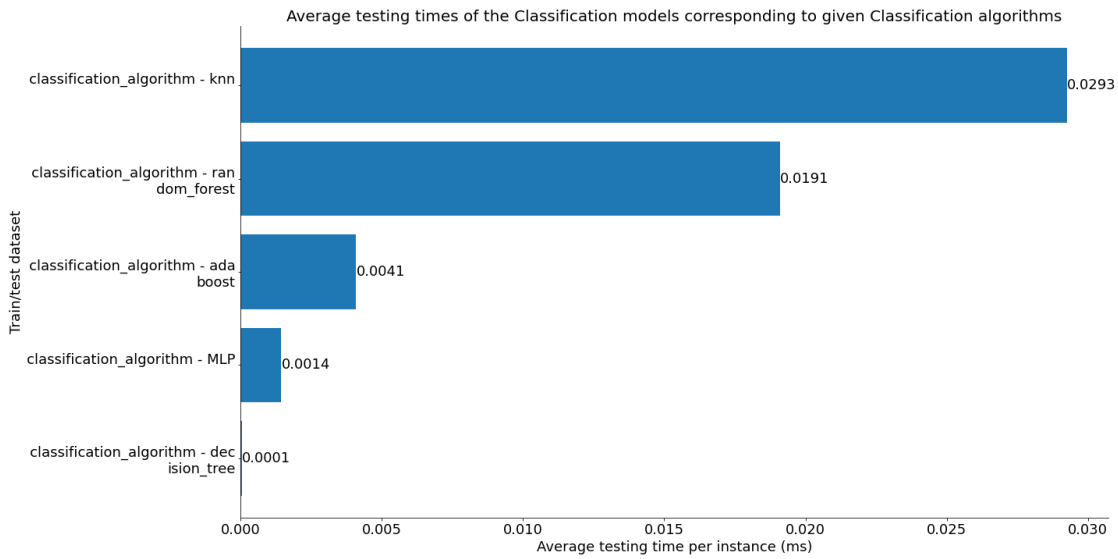

Figure 18. Average testing times of classification models corresponding to given classification algorithms

Looking at the times, the decision tree model was able to classify the data record multiple times quicker than the other methods. While that model did not achieve the overall highest average f1-score, it did achieve the second-best results. The most accurate model, random forest, resulted in some of the slowest predictions. This makes sense, since random forest

classifier is essentially a collection of decision trees, while the quickest model consists of just a single tree. While it is possible, that in some cases the single decision tree might make mistakes and not be able to correctly predict as many cases as the random forest model, then in the current tests both models resulted in very similar results and because of this the initial selection will be made mainly based on the testing times. This means, that decision tree can be fixed as the classification model.

After analysing the average results across the tests, the fixed values that the comparison of different training and testing datasets will be carried out on has been chosen. The fixed values are shown in Table 40.

Table 40. Selected fixed values for training and testing dataset analysis

Feature selection algorithm	Fisher's score
Feature selection dataset	Benign-gafgyt-mirai-balanced
Power of the feature set	10
Classification algorithm	Decision tree

4.2.2 Comparison of classification models trained and tested on different devices

With all other parameters other than the training and testing dataset have been fixed the analysis for those devices can begin. The list of training and testing datasets that will be analysed has already been described in paragraph 3.2.2. Just like with anomaly detection, a total of 25 different models will be compared.

4.2.2.1 Classification results

After 25 different models has been selected, the results of all these models can then be compared. On Table 41 the accuracies, f1-scores, and testing times per instance for each model are shown. Since a single decision tree model is very fast, then the testing times per instance are miniscule. Because of this, for the following tests, the times are now shown in microseconds instead of the previous milliseconds. The results are also illustrated on Figure 19.

Table 41. Results for Decision Tree classification models trained and tested on different devices

Testing Device	Category	Model trained on all-devices			Model trained on different device categories (cameras, doorbells)			Model trained on single device		
		Accuracy	F1-score	Testing time (µs)	Accuracy	F1-score	Testing time (µs)	Accuracy	F1-score	Testing time (µs)
Provision PT-737E	Cameras	0.95647	0.95157	0.05553	0.98790	0.98793	0.06142	0.99832	0.99833	0.03801
Provision PT-838	Cameras	0.96050	0.95660	0.05589	0.99376	0.99376	0.06762	0.99973	0.99973	0.03176
SimpleHome XCS7-1002-WHT	Cameras	0.99897	0.99897	0.05895	0.99533	0.99532	0.05537	0.99888	0.99888	0.03735
SimpleHome XCS7-1003-WHT	Cameras	0.99793	0.99793	0.05755	0.99780	0.99780	0.05420	0.99933	0.99933	0.03181
Samsung SNH 1011N[5]	Cameras	0.99510	0.99510	0.08410	0.99635	0.99640	0.07315	0.99911	0.99910	0.04728
Ennio doorbell[5]	Doorbells	0.99660	0.99660	0.07975	0.93700	0.93280	0.07455	0.99968	0.99970	0.03586
Danmini doorbell	Doorbells	0.98527	0.98511	0.05368	0.76370	0.73041	0.06464	0.99874	0.99873	0.03754
Ecobee thermostat	-	0.99887	0.99887	0.05491	-	-	-	0.99944	0.99944	0.03106
Philips B120N/10	-	0.98517	0.98485	0.05519	-	-	-	0.99961	0.99961	0.03161
Average		0.98681	0.98578	0.05611	0.95016	0.94626	0.06232	0.99920	0.99921	0.03581

[5] No Mirai data available

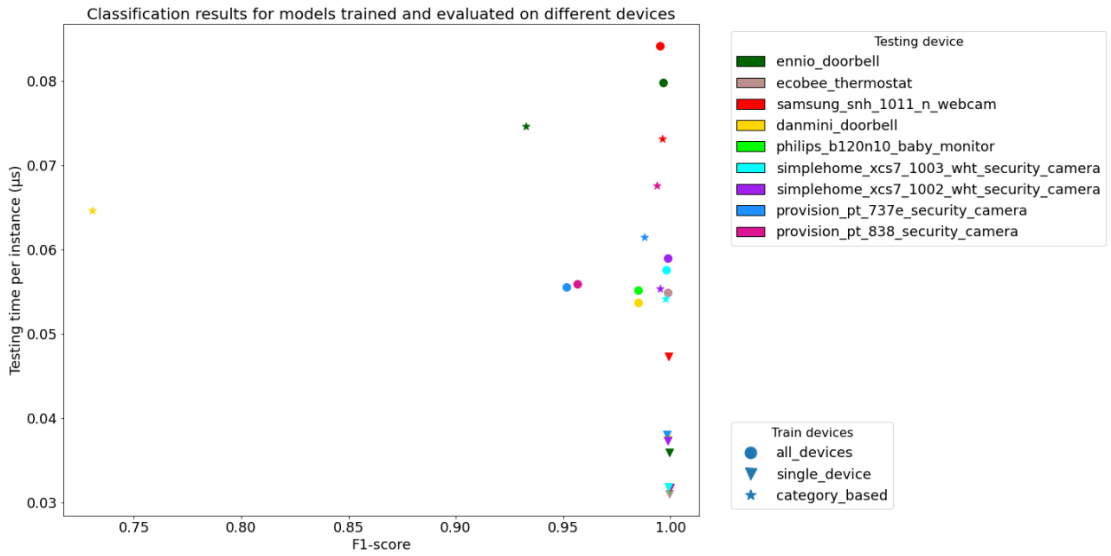

Figure 19. f1-scores and average testing times of different classification models

Looking at the results, most of the models were able to correctly classify more than 99% of the records, regardless of the dataset they were trained on. However, there were still a couple of anomalies within the models, where a model achieved lower accuracies and f1-scores than the rest. One of the biggest differences happened with the category-based models that were evaluated on either type of doorbell. The models tested on Ennio doorbell and Danmini doorbell achieved f1-scores of about 93.6% and 76.4% respectively. When models that were either trained on all devices or just a single device were evaluated against the same doorbell data, then the models did much better and got results of around 99%. Other than that, the only other models that classified records with lesser accuracy were models that were trained on all devices and evaluated against data from either of the Provision security cameras. These models both achieved f1-scores of around 95%.

To see why the doorbells models achieved such low results when trained on both types of doorbell data compared to other models, then a closer look to a few of the confusion matrixes will be taken. In Table 42 and Table 43, the confusion matrices for models trained on doorbells data and Danmini doorbell data respectively and evaluated on Danmini doorbell data are shown.

Table 42. Confusion matrix for classification model trained on doorbells data and evaluated on Danmini doorbell data

	Predicted benign	**Predicted Gafgyt**	**Predicted Mirai**
Actual benign	8342	66	1592

Actual Gafgyt	6	8563	1431
Actual Mirai	4	3990	6006

Table 43. Confusion matrix for classification model trained on Danmini doorbell data and evaluated on Danmini doorbell data

	Predicted benign	**Predicted Gafgyt**	**Predicted Mirai**
Actual benign	73264	115	8
Actual Gafgyt	117	73496	21
Actual Mirai	2	16	73679

From the doorbells model confusion matrix, when comparing just the benign and attack records, the model did well in correctly detecting attack data from normal data with only 10 attack records out of a total 20000 got incorrectly classified as benign. On the other side however, approximately 16.5% (1658 out of 10000) of the normal records did get incorrectly classified as attack, with most of them getting labeled as Mirai. Looking at the two types of attack records, the model did have a hard time in differentiating them with about 14.3% (1431 out of 10000) of Gafgyt data getting classified as Mirai and 39.9% (3990 out of 10000) of Mirai data getting incorrectly classified as Gafgyt. From the single device model confusion matrix on Table 43, the model was able to correctly classify all three types of data with high accuracy. The single device model did make more mistakes when classifying between benign and Gafgyt data, unlike the doorbells model, where less mistakes were made between benign and Gafgyt data and the low accuracies came mainly because of Mirai data. This could be explained by the fact that additionally to Danmini doorbell, the doorbells model had data from just one other device, Ennio doorbell, which did not have any Mirai data available for it. This means that during training the model had less Mirai data available and might have suffered from undertraining.

4.2.2.2 Classification models performance

Additionally, to the accuracies and f1-scores, testing times for the models need to be taken into consideration as well. Looking at Table 43, most of the models that were trained on multiple devices managed to predict the cases in a similar time-frame of around 0.05-0.06 μs per instance, regardless of the amount of different devices the combined training set was sampled from. However, the models did perform faster when just training the model on a single device, with most of them taking around 0.03-0.04 μs per instance. Additionally to that, similarly to the previously analysed anomaly detection models, the

models that were tested on the two different devices that did not have any Mirai data available achieved the slowest times of around 0.07-0.08. Overall, all the models performed exceptionally quickly and could possibly all be great choices for real time applications. Main reason for this is the selected Decision tree model and the results might vary when using a different classification method.

4.2.3 Models summary

The extensive tests of different training and testing sets showed that most of the models worked well and were able to correctly classify more than or around 99% of the data. A more detailed look at the numbers shows that in most cases the models that were trained on single device managed to achieve slightly higher results than the rest. In addition to that, the single device models achieved the result on average 30%-40% faster than the more complex models. Regardless of that, all the models were still very time effective and the differences in accuracies and f1-scores were not large, meaning that depending on the use case, combined models might be preferred to reduce the maintenance cost.

4.3 Feature selection dataset analysis

The current thesis proposed two main questions that needed to be answered, to analyse combined models of multiple devices and to see if having prior knowledge of attack data does affect the results of feature selection. While the second point is especially important for unsupervised methods, which does not require for any attack data to be previously available, then the same goal will be investigated for supervised methods as well.

4.3.1 Choosing the fixed values

Most of the values are fixed based on the average results that were analysed in paragraph 3.2.1. The training dataset device and testing dataset device will be selected based on the results seen in Table 41. Since most of the models tested in the previous analysis performed similarly, then the selection is not quite as straightforward. On account of uniformity, the same training and testing devices as in the anomaly detection analysis will be selected, meaning that both the training and testing data samples will be taken from the SimpleHome XCS7-1003-WHT data. The chosen fixed values are shown in Table 44.

Table 44. Selected fixed values for feature selection dataset analysis for classification

Training device	SimpleHome XCS7-1003-WHT

Testing device	SimpleHome XCS7-1003-WHT
Feature selection algorithm	Fisher's score
Power of the feature set	10
Classification algorithm	Decision Tree

4.3.2 Comparison of classification models with different feature selection datasets

With the other parameters given fixed values, the comparison and analysis of models where features were selected from different datasets can begin. Just like in section 3.2.2, the data will be divided into 5 different datasets and the same series of feature selection and classification methods will be carried out on the data.

4.3.2.1 Benign data

Here, the features were chosen from strictly benign data. The 10 best features that were selected by Fisher's score from the data can be seen in Table 45.

Table 45. Selected features for benign data by Fisher's score

MI_dir_L5_weight
MI_dir_L5_mean
MI_dir_L5_variance
MI_dir_L3_weight
MI_dir_L3_mean
MI_dir_L3_variance
MI_dir_L1_weight
MI_dir_L1_mean
MI_dir_L0.1_weight
MI_dir_L0.1_mean

From Table 45, all of the features from benign data that achieved the highest Fisher's score belonged to the source MAC-IP (MI) category. The times varied a lot with every time-frame, other than the largest 1 min time-frame, getting selected at least twice. However, the method still seemed to prefer smaller time-frames with 100ms and 500ms getting chosen for three features.

After the features had been selected, the Decision tree model was trained and evaluated to classify the records with now less features. The classification model results and the confusion matrix are seen in Table 46 and Table 47, respectively.

Table 46. Results for classification model with 10 features selected by Fisher's score for benign data

Accuracy	**Precision**	**Recall**	**F1-score**	**Testing time per instance (µs)**	**Training time per instance (µs)**
0.99933	0.99932	0.99933	0.99932	0.46776	0.04277

Table 47. Confusion matrix for classification model with 10 features selected by Fisher's score for benign data

	Predicted benign	**Predicted Gafgyt**	**Predicted Mirai**
Actual benign	73503	90	1
Actual Gafgyt	39	73334	2
Actual Mirai	8	9	73821

The Decision tree model was able to correctly classify over 99.9% of the records from a dataset where the best features were chosen by Fisher's score from benign data only. From the confusion matrix, the model misclassified just 149 records out of the total 220807 data points.

4.3.2.2 Benign and Gafgyt data balanced

Here, the features were chosen from a mixed data of benign and Gafgyt records with equal amounts of both types of data. The 10 best features that were selected by Fisher's score from the data can be seen in Table 48.

Table 48. Selected features for balanced benign and Gafgyt data by Fisher's score

MI_dir_L0.01_variance
H_L5_weight
H_L5_mean
H_L5_variance
H_L3_weight
H_L0.01_variance
HH_L5_weight
HH_L5_mean
HH_L5_std
HH_L5_magnitude

After adding Gafgyt data to the feature selection dataset, Fisher's score resulted in highly different results. Unlike with the benign only data, this time a feature from the Source MAC-IP category was selected just once. Every other feature was chosen from either the Source IP or Channel category. Time wise the model clearly preferred smaller time-frames for the most part with 7 out of 10 features coming from the smallest 100ms time-frame. Contrary to that however, two features also got selected from the largest time-frame.

The classification model results and the confusion matrix are seen in Table 49 and Table 50, respectively.

Table 49. Results for classification model with 10 features selected by Fisher's score for benign and Gafgyt balanced data

Accuracy	**Precision**	**Recall**	**F1-score**	**Testing time per instance (µs)**	**Training time per instance (µs)**
0.99890	0.99890	0.99890	0.99890	0.45932	0.04610

Table 50. Confusion matrix for classification model with 10 features selected by Fisher's score for benign and Gafgyt balanced data

	Predicted benign	**Predicted Gafgyt**	**Predicted Mirai**
Actual benign	73916	102	0
Actual Gafgyt	63	73253	15
Actual Mirai	8	56	73512

From Table 49, after adding Gafgyt data to the feature selection dataset, the model became slightly less accurate. This time the Decision tree model achieved accuracy and f1-score of little under 99.9%. However, the accuracy was still high and only 244 records out of the total 220925 points got classified incorrectly.

4.3.2.3 Benign and Gafgyt data unbalanced

Here, the features were chosen from a mixed data of benign and Gafgyt records with more Gafgyt records available. The 10 best features that were selected by Fisher's score from the data can be seen in Table 51.

Table 51. Selected features for unbalanced benign and Gafgyt data by Fisher's score

HH_L5_weight
HH_L5_std
HH_L5_radius
HH_L5_pcc
HH_L3_weight
HH_L3_mean
HH_L3_std
HH_L3_magnitude
HH_L3_radius
HpHp_L0.01_pcc

After biasing the dataset more towards Gafgyt data, the feature selection model started choosing features mainly from the Channel category with just one feature additionally getting selected from the Socket category. Time wise the results did not differ too much, and most features still came from smaller time-frames. The classification model results, and the confusion matrix are seen in Table 49 and Table 50, respectively.

Table 52. Results for classification model with 10 features selected by Fisher's score for benign and Gafgyt unbalanced data

Accuracy	**Precision**	**Recall**	**F1-score**	**Testing time per instance (µs)**	**Training time per instance (µs)**
0.85096	0.89623	0.85096	0.84870	0.43216	0.04385

Table 53. Confusion matrix for classification model with 10 features selected by Fisher's score for benign and Gafgyt unbalanced data

	Predicted benign	**Predicted Gafgyt**	**Predicted Mirai**
Actual benign	69210	3266	1088
Actual Gafgyt	90	66166	7155
Actual Mirai	47	21247	52445

With less benign data available during feature selection, the model achieved much worse results than with the previous two feature selection datasets. This time the model got accuracy and f1-score of approximately 85% compared to the near perfect scores of the previous models. Looking at the confusion matrix at Table 53, it appears that while the

model still did a decent job in correctly predicting benign data, where 99.8% (69210 out of 69347) of the records that got classified as benign were from benign data, then it failed to correctly classify the attack records. Overall, 32893 records out of the total of 220714 cases were predicted incorrectly.

4.3.2.4 Benign, Gafgyt and Mirai data balanced

Here, the features were chosen from a mixed data of benign, Gafgyt and Mirai records with equal amounts of all types of data. The 10 best features that were selected by Fisher's score from the data can be seen in Table 54.

Table 54. Selected features for balanced benign, Gafgyt and Mirai data by Fisher's score

MI_dir_L0.01_variance
H_L5_weight
H_L5_mean
H_L0.01_weight
H_L0.01_mean
H_L0.01_variance
HH_L5_weight
HH_L5_mean
HH_L5_std
HH_L5_magnitude

With the addition of Mirai data, the feature selection method achieved very similar results to the balanced Benign and Gafgyt data balanced at 4.3.2.2. The selected features all came from the same categories and with similar time-frames. This time the selected features all belonged either to the smallest time-frame of 100ms or the largest time-frame of 1m.

The Decision tree model results and the confusion matrix can be seen in Table 55 and Table 56.

Table 55. Results for classification model with 10 features selected by Fisher's score for benign, Gafgyt and Mirai balanced data

Accuracy	**Precision**	**Recall**	**F1-score**	**Testing time per instance (µs)**	**Training time per instance (µs)**
0.99933	0.99933	0.99933	0.99933	0.38905	0.03180

Table 56. Confusion matrix for classification model with 10 features selected by Fisher's score for benign, Gafgyt and Mirai balanced data

	Predicted benign	**Predicted Gafgyt**	**Predicted Mirai**
Actual benign	73604	101	6
Actual Gafgyt	15	73677	3
Actual Mirai	3	20	73328

With a balanced dataset of benign, Gafgyt and Mirai records, the model achieved similar results to the benign only model. The accuracy and f1-score of the model are both over 99.9%. From the confusion matrix, the model misclassified just 148 records out of the total 220757 data points.

4.3.2.5 Benign, Gafgyt and Mirai data unbalanced

Here, the features were chosen from a mixed data of benign, Gafgyt and Mirai records with more Gafgyt and Mirai records available. The 10 best features that were selected by Fisher's score from the data can be seen in Table 57.

Table 57. Selected features for unbalanced benign, Gafgyt and Mirai data by Fisher's score

HH_L5_weight
HH_L5_std
HH_L5_radius
HH_L5_pcc
HH_L3_weight
HH_L3_mean
HH_L3_std
HH_L3_magnitude
HH_L3_radius
HpHp_L0.01_pcc

With the last feature selection dataset, where all three types of data were present, but the data was more biased towards attack data, Fisher's score selected exactly the same best features than it did with the unbalanced benign and Gafgyt dataset at 4.3.2.3.

The classification model results and the confusion matrix are seen on Table 58 and Table 59, respectively.

Table 58. Results for classification model with 10 features selected by Fisher's score for benign, Gafgyt and Mirai unbalanced data

Accuracy	Precision	Recall	F1-score	Testing time per instance (µs)	Training time per instance (µs)
0.85042	0.89566	0.85076	0.84821	0.51478	0.04293

Table 59. Confusion matrix for classification model with 10 features selected by Fisher's score for benign, Gafgyt and Mirai data

	Predicted benign	Predicted Gafgyt	Predicted Mirai
Actual benign	69306	2879	1546
Actual Gafgyt	95	63588	10090
Actual Mirai	46	18362	54827

Since the selected features were the same as they were in 4.3.2.3, then the overall accuracies and f1-scores did not differ by a large margin either and the model was able to correctly predict about 85% of the records. Slight differences did show up because of the randomness of the bootstrap method with 33018 records out of the total 220739 cases getting predicted incorrectly.

4.3.3 Models summary

In Table 60, the distribution of the best 10 features selected by different feature selection datasets according to their categories can be seen.

Table 60. Distribution of the best 10 features selected from different feature selection datasets

Category	Benign	Benign, gafgyt balanced	Benign, gafgyt unbalanced	Benign, gafgyt, mirai balanced	Benign, gafgyt, mirai unbalanced
Source IP		5		5	
Source MAC-IP	10	1		1	
Channel		4	9	4	9
Channel jitter					
Socket			1		1

As was discovered during the analysis, the addition of Mirai data did not affect the outcome of Fisher's score feature selection method too much. Both the models based on the benign and Gafgyt as well as the benign, Gafgyt and Mirai datasets, selected features from the same categories. However, when comparing the selected features with the benign only model, then bigger differences can be seen. While from just benign data, all the most discriminating features came from the source MAC-IP category, then with the addition of attack data, the selected features started curving more towards other categories. With the balanced datasets, the features got mainly chosen from source IP and channel categories and with datasets more biased towards attack data, most of the features came from the channel category.

The overall results of the five different models can be seen in Table 61. All the models were trained and evaluated on the same data with the only difference being the features the datasets had available. From the table, two distinct levels of models show up. The three models from which the features were selected from a balanced dataset achieved considerably higher results than the models where the dataset consisted of more attack data and less benign data. While the differences between the models based on the balanced datasets were not huge, then the benign only and the benign, Gafgyt, Mirai models were able to correctly classify just slightly more records than the benign and Gafgyt based model, of around 99.93% compared to 98.9% respectively. These results support the findings of [6], where it was discovered that the host- or source-based feature categories had the highest discriminatory powers. From the testing times, the classification model where the features were selected from a balanced dataset of all three types of data, managed to perform just slightly quicker than the rest.

Table 61. Results for classification models with 10 features selected by Fisher's score from different data sample sets

Feature selection data sample set	**Accuracy**	**F1-score**	**Testing time per instance (µs)**
Benign	0.99933	0.99932	0.46776
Benign, gafgyt balanced	0.99890	0.99890	0.45932
Benign, gafgyt unbalanced	0.85096	0.84870	0.43216
Benign, gafgyt, mirai balanced	0.99933	0.99933	0.38905

Benign, gafgyt, mirai unbalanced	0.85042	0.84821	0.51478

4.4 Feature selection algorithm analysis

One part of the third, slightly smaller goal was to compare different feature selection algorithms for both supervised and unsupervised learning methods.

4.4.1 Choosing the fixed values

As it has been done with every other part of the analysis, first all parameters need to be given a fixed value. Most of the values, other than the feature selection dataset are taken from parts 4.2.1 and 4.3.1. Feature selection dataset selection is based on the results of 4.2. From the analysis, it was found that the models which used feature selection datasets of just benign data and of all three types of data in balanced form achieved the best overall results. However, since benign-gafgyt-mirai-balanced data was already used during anomaly detection part of the analysis, then it will be selected here as well. The chosen fixed values are shown in Table 62.

Table 62. Selected fixed values for classification algorithm analysis

Training device	SimpleHome XCS7-1003-WHT
Testing device	SimpleHome XCS7-1003-WHT
Feature selection dataset	Benign-gafgyt-mirai-balanced
Power of the feature set	10
Classification algorithm	Decision tree

4.4.2 Comparison of classification models with different feature selection algorithms

After the other parameters have been given fixed values, the analysis of how different feature selection algorithms can affect the outcome of feature selection begins. In total, 10 features are chosen by 5 different feature selection algorithms and the results of the output models are compared. Since four out of the five algorithms were already analysed more deeply in section 3.4.2 and the Fisher's score output with the currently fixed parameters was already analysed in section 4.3.2.2, then the individual analysis of each model will be skipped here.

4.4.3 Models summary

In Table 63, all the features that were selected by different feature selection algorithms can be seen and in Table 64, the distribution of the best 10 features are shown.

Table 63. Features chosen by different feature selection algorithms for classification

Entropy	**Variance**	**Gini index**	**Hopkins statistic**	**Fisher's score**
HH_L1_weigh t	HH_jit_L5_me an	MI_dir_L5_me an	HH_L5_radius	MI_dir_L0.01 _variance
HH_L1_std	HH_jit_L5_var iance	MI_dir_L0.1_ mean	HH_L5_covari ance	H_L5_weight
HH_L1_radius	HH_jit_L3_me an	MI_dir_L0.01 _mean	HH_L3_radius	H_L5_mean
HH_L0.1_wei ght	HH_jit_L3_var iance	HH_L0.01_me an	HH_jit_L1_var iance	H_L0.01_weig ht
HH_L0.1_std	HH_jit_L1_me an	HH_L0.01_ma gnitude	HH_jit_L0.01_ mean	H_L0.01_mea n
HH_L0.1_radi us	HH_jit_L1_var iance	HH_jit_L5_me an	HpHp_L3_ma gnitude	H_L0.01_varia nce
HH_L0.01_std	HH_jit_L0.1_ mean	HH_jit_L3_me an	HpHp_L3_cov ariance	HH_L5_weigh t
HH_L0.01_rad ius	HH_jit_L0.1_v ariance	HH_jit_L1_me an	HpHp_L1_ma gnitude	HH_L5_mean
HH_jit_L1_we ight	HH_jit_L0.01_ mean	HH_jit_L0.1_ mean	HpHp_L0.1_m ean	HH_L5_std
HH_jit_L1_var iance	HH_jit_L0.01_ variance	HH_jit_L0.01_ mean	HpHp_L0.01_ magnitude	HH_L5_magni tude

Table 64. Distribution of the best 10 features selected by different feature selection algorithms

Category	**Entropy**	**Variance**	**Gini index**	**Hopkins statistic**	**Fisher's score**
Source IP					5
Source MAC-IP			3		1
Channel	8		2	3	4
Channel jitter	2	10	5	2	
Socket				5	

From the tables, a specific difference between the unsupervised feature selection methods and the supervised Fisher's score method can be seen. While for most of the unsupervised

algorithms the features got mainly selected from the channel and channel jitter categories with a few aberrations, then Fisher's score was more biased towards the host-based categories. From the unsupervised methods, Gini index additionally selected 3 features from the source MAC-IP category and Hopkins statistic selected 5 features from the Socket category. From Table 63, time wise there were no very clear preferences in almost any of the methods. While variance, Gini index and Hopkins statistic selected features from every time-frame available, then entropy preferred mainly the larger time-frames and Fisher's score both the smallest and the largest time-frames.

The overall results of the different models can be seen in Table 65. From the table, most of the models were able to correctly classify over 97.5% of the data records with the only exception being the model where the features were selected by entropy. The entropy-based model achieved an f1-score of only about 86%. Not only did the model perform considerably worse than the rest, but it also took the longest to make the predictions. From the other models, two methods stand out. These methods are the Fisher's score based model, which classified over 99.9% of the records correctly and the Gini index based model, which was able to achieve even higher results than Fisher's score. Both models also took the least amount of time during testing. Since both of the best performing feature selection algorithms were the only two that also selected some features from the source-based categories, then that also confirms the findings in paragraph 4.3.2 and in [6], that the host- or source-based categories have the highest discriminatory powers.

Table 65. Results for classification model with 10 features selected by different feature selection algorithms

Feature selection method	**Accuracy**	**F1-score**	**Testing time per instance (µs)**
Entropy	0.86472	0.86017	0.73848
Variance	0.97679	0.97607	0.63495
Gini index	0.99948	0.99948	0.48106
Hopkins statistic	0.98492	0.98463	0.56298
Fisher's score	0.99933	0.99933	0.38905

4.5 Power of the feature set and classification algorithm analysis

For the last part of the classification analysis, both two remaining parameters, power of the feature set and anomaly detection algorithm will be looked at together.

4.5.1 Choosing the fixed values

Similarly to anomaly detection part of the analysis, every combination of feature set powers and classification algorithms will be analysed. Since the tests were ran with three different feature set sizes of 3, 5 and 10 and five different classification algorithms, then in total 15 different models will be analysed. Every other parameter will receive their fixed value from the previous tests. The chosen fixed values are shown in Table 66.

Table 66. Selected fixed values for feature set powers and classification algorithm analysis

Training device	SimpleHome XCS7-1003-WHT
Testing device	SimpleHome XCS7-1003-WHT
Feature selection dataset	Benign-gafgyt-mirai-balanced
Feature selection algorithm	Fisher's score

4.5.2 Comparison of classification models with different feature set powers and classification algorithms

For the feature set powers and the classification methods analysis, first from the sample set 3, 5 or 10 best features are selected based on the results of the chosen feature selection algorithm. The training and testing datasets are then reduced such that they only contain these selected features. These sets are then used to train and evaluate the particular classification models. In Table 67, the results of each power of the feature set and classification method combination can be seen.

Table 67. Classification results for different feature set powers and classification algorithms

Feature set power	Classification algorithm	Accuracy	Precision	Recall	F1-score	Training time per instance (µs)	Testing time per instance (µs)
3	Decision tree	0.98217	0.98381	0.98222	0.98204	0.40755	0.07458
5	Decision tree	0.98559	0.98685	0.98558	0.98549	0.34533	0.04688
10	Decision tree	0.99933	0.99933	0.99933	0.99933	0.38905	0.03181
3	Random forest	0.99089	0.99105	0.99090	0.99089	118.625	17.658
5	Random forest	0.99165	0.99183	0.99165	0.99165	128.043	15.834

10	Random forest	0.99941	0.99941	0.99941	0.99941	195.617	14.673
3	KNN	0.99449	0.99452	0.99449	0.99449	1.9957	23.0596
5	KNN	0.99809	0.99809	0.99809	0.99808	2.3268	24.4457
10	KNN	0.99223	0.99235	0.99221	0.99223	2.0799	29.9754
3	Adaboost	0.86299	0.87913	0.96321	0.86192	29.4285	4.0093
5	Adaboost	0.88470	0.89520	0.88434	0.88257	31.0801	3.4887
10	Adaboost	0.92389	0.93507	0.92380	0.91362	51.7763	3.6783
3	MLP	0.86290	0.88626	0.96282	0.85500	63.7428	1.2550
5	MLP	0.92161	0.93387	0.92152	0.92065	62.6586	1.3077
10	MLP	0.98386	0.98515	0.98388	0.98379	50.4256	1.1012

Looking at the results in Table 67, most of the methods achieved high 98%-99% accuracies and f1-scores with all of the feature set sizes. In the most part, having more features did cause the models to perform more accurately with the only exception of KNN model where the 10-feature model got slightly worse results than the 3 or 5 feature counterparts. The difference was not huge however, with the 10-feature model averaging at f1-score of 99.2% and 3 and 5 feature models at 99.4% and 99.8% respectively.

When comparing the results classification algorithm wise, then two methods stand up from the rest, scoring moderately lower accuracies and f1-scores than the rest. For both Adaboost and the MLP classifier the 3-feature model was only able to correctly classify approximately 86% of the records compared to other methods 98%-99%. With higher feature set powers, the Adaboost and MLP models did get more accurate, achieving f1-scores of 91.4% and 98.4% respectively, which however are still lower than for the rest of the methods. The results for the other methods stayed similar to each other with 10-feature Random Forest model achieving the overall highest f1-score of 99.94%.

Comparing the average times the different models took, the testing times do correlate with the overall average results found at Figure 18. The KNN algorithm took the longest to evaluate with all three feature set sizes, followed by Random Forest. By a large magnitude, the fastest algorithm was the single Decision tree working multiple hundreds of times quicker than the slowest KNN. It should be mentioned that the timings might vary a lot based on the parameters set for the models, such as the number of neighbors for KNN or the number of trees for Random Forest. In the current thesis, these parameters were set experimentally during the tuning phase of a model. For example, the current

Random forest classifier consists of a total of 400 trees and the KNN bases the predictions on 14 neighbors. Reducing these numbers could help speed up the models, without possibly affecting the results too much. While the KNN model achieved the worst timings during the testing phase, then it was the second fastest model to train, following Decision tree. The reason for this is the aforementioned *lazy learning* of KNN. The slowest method by far was Random forest which took multiple times longer to train than the second slowest MLP classifier. This again was mainly caused by the large number of trees the method consisted of and could possibly be reduced quite a bit.

The results show that for effective IoT botnet prediction a variety of different classification methods can be used. Three of the five tested models provided effective results even with the smallest amounts of features and one model started getting similar accuracies to others with larger number of features. The only method from the current tests that achieved slightly lower results was Adaboost, but even that was able to correctly classify more records than the majority of the unsupervised methods. From the performance part, a single Decision tree provided to achieve fairly accurate results very fast and could be a great candidate for IoT botnet detection. Random forest was able to reduce the incorrectly classified results of a single Decision tree, but made the model take a longer time. Therefore, the Random forest classifier with smaller forest size than was currently tested could also be a good choice for IoT botnet detection.

5 Discussion

In the study, many different supervised and unsupervised botnet detection models were trained, and their results analysed. The first part of the analysis covered finding the parameters that result in the overall best performing anomaly detection models and the second half of the thesis did the same for classification models. One of the main goals of the thesis was to find out how much does the presence of attack data during the feature selection phase of botnet detection models affect the features that get chosen and the accuracy of the anomaly detection and classification models. During anomaly detection analysis, Hopkins statistic was used to select the best features. Comparing the features that got selected from just benign data with features that got selected from benign and Gafgyt or benign, Gafgyt and Mirai data, showed that on the most part the best performing features stayed the same. The main difference was that with data which had higher percentage of attack data presence, more features got selected from channel category and less from channel jitter. The resulting anomaly detection models showed that while models where features were selected from benign only or from benign and Gafgyt data achieved similar results, then with the addition of Mirai data, the accuracies and f1-scores went up by a considerable margin. Testing times stayed similar along all the tests.

The second goal was to compare models that were trained on just a single device with models that are trained on more than one machine. For this, three different models with the same parameters were trained for each device, one that was trained on the single device, one that was trained on multiple devices belonging to the same category and one that was trained on all devices. The result of the analysis was slightly unexpected, as for the most part, the combined models performed more accurately than models that were trained on just a single device. There were some exceptions, but this was true for the majority. The combined models also did so without negatively affecting the performance.

Lastly, the thesis also compared the different remaining parameters of the models: the feature selection algorithms, powers of the feature set and anomaly detection and classification algorithms. For anomaly detection, the models which selected features based on the results of Gini index averaged in the most accurate results. It was followed

by Hopkins statistic and Variance. The worst performing models by a large margin used entropy as a mean of selecting features. From the list of features that every model selected, distinct feature selection methods selected the features from different feature categories. While Entropy and Hopkins statistic selected most of the features from the channel and channel jitter categories, then Variance chose features only from jitter. Gini index was the only one which skipped the jitter category and additionally took features from the host-based categories, indicating higher discriminatory powers of these categories.

For selecting the best anomaly detection algorithm, three methods were looked at: One-Class SVM, Isolation Forest and Local Outlier Factor. The analysis showed clear results of the Isolation Forest performing much better than the other two methods. The highest f1-score for IF was around 89.8% compared to 67% for SVM and 67.8% for LOF. In addition, IF was a lot faster than the others. The different powers of the feature sets had a unique effect to each different method. While for SVM not much changed regardless of the amount of features the data had, with LOF the training and testing times went up as more features were present. With the best performing IF method the 5-feature model resulted in the highest f1-score.

After completing the analysis for the unsupervised anomaly detection methods, the same points were covered for the supervised classification models as well. During the feature selection dataset analysis, Fisher's score was used to select the best features. With Fisher's score most of the features that were selected from either the benign only dataset or either of the balanced datasets belonged to the host-based categories. For the remaining two, biased towards attack data datasets, all of the features were chosen from channel and socket categories. Additionally, it was seen that the addition of Mirai data did not affect the feature selection this time and the chosen features stayed the same as with benign and Gafgyt datasets. Overall, the classification models where features were selected from either benign only or balanced dataset were all able to correctly classify around 99.9% of the data records without differing too much from the performance perspective either. The models based on the unbalanced sets performed much worse of around 85% f1-score, indicating once more that the host-based features have higher discriminatory powers.

When comparing models trained on a single device and models trained on multiple devices, then most of the resulting models were able to achieve high accuracy results regardless of the dataset used. However, this time for the most part the single device

models did prove to be slightly more effective than the rest. These results were largely based on the device they were tested on, for some the difference between all-devices model and a single device model was miniscule and all the models were able to correctly predict over 99% of the records, while for others the single device model resulted in reasonably higher accuracies and f1-scores. Additionally, on the performance part, the single device models were able to classify records almost twice as fast as the other models.

When comparing the feature selection algorithms, a similar trend to some of the previous results were seen. Out of the five tested feature selection methods, two of them, Gini index and Fisher's score, selected at least some of the features from the host-based categories. The classification models based on those two methods were also the most accurate, scoring f1-scores of over 99%. Following them were the Hopkins statistic, variance, and entropy, which based models classified 98.5%, 97.6% and 86% of the records correctly, respectively. The two most accurate models were also the quickest.

Unlike with anomaly detection, this time the feature set powers had an effect on almost all of the models results. For the most part, the higher feature set sizes were also able to correctly classify more records correctly without negatively affecting the performance. From the five tested classification algorithms, the Decision tree, Random forest and KNN models all achieved f1-scores of around 99.9%. From these, the decision tree models were by far the quickest, averaging about 350 times lower times than the next fastest random forest method.

Overall, the tests showed that just having extra Gafgyt data available during feature selection compared to only benign data, does not affect the supervised and unsupervised learning models by a lot. However, after selecting the features based on all three types of data, the IF anomaly detection algorithm did get more accurate than the rest while for supervised classification the results stayed similar to the previous models.

Additionally, the tests showed that it is possible to create common models for multiple devices without losing much accuracy. For anomaly detection, the more complex models proved to be even more accurate than the models trained on single devices while for classification the combined models did achieve slightly lower scores but depending on the device the model was tested against, these differences were often miniscule. However,

it was also seen, that these results can differ from device to device, meaning that this conclusion might not be true to every single IoT device.

Lastly, for the feature selection, Hopkins statistic and Gini index were able to get the best results for anomaly detection and Gini index and Fisher's score for classification. For the anomaly detection algorithm, isolation forest was able to perform much more accurately and much quicker than the rest, regardless of the number of features that were present. For classification, decision tree was by far the quickest to perform and achieved very high results for 10-feature model. If testing time is not as important of a factor, then both KNN and Random Forest were also able to get f1-scores of over 99.9% with longer times.

6 Future work

While the thesis did cover many different sides of supervised and unsupervised IoT botnet detection models, there is a lot of future analysis that can be written on the topic. First, additionally to the results analysed in the thesis, many extra tests were carried out that were not deeply covered in the study. These tests covered all kinds of possible combinations of different methods and because of this the current analysis might not reflect the overall best models or results that work for every situation. The test results are all provided in Appendix 2 and future analysis can be based on the findings.

In addition to the used models in the current analysis, more work can also be put into analysing deep learning and neural network models. The current thesis only touched the surface of the topic by analysing the MLP classifier but did not dive more deeply into it. While deep learning has been analysed and used in previous works, such as the bachelor thesis written in TalTech [43], which centered around using neural networks for IoT botnet detection, then to my knowledge, it has not been as extensively researched.

On the power of the feature set size, the thesis only worked with features of size 3, 5 and 10. While these sizes do not cover all the possibilities, then they did offer an overlook of some of the different magnitudes. More work could be put into analysing the different powers and narrowing the selection down to the absolute best windows of values, which give out the most accurate and quickest results.

Lastly, currently the classification part of the analysis only differentiated between three types of data: benign, Gafgyt and Mirai. In addition to this, the initial data was divided into different attack types, as described in section 2.1. It could be useful to expand the analysis further in the future to be able to additionally classify the records based on what type of attack it tries to carry out, as that might help in deciding what protective measures need to be taken.

7 Conclusion

In the present thesis, an extensive analysis of thousands of different supervised and unsupervised IoT botnet detection models was carried out and highly promising results were seen. The analysis showed that with the use of feature selection it is possible to reduce the complexity of the dataset and the overall cost of botnet detection models while still maintaining highly accurate models. It was shown that while the selected features did differ depending on how much benign, Gafgyt and Mirai was initially available, then for the most part less complex dataset models were still able to produce substantial results for both supervised and unsupervised methods.

Secondly, the paper did find out that creating combined models that are trained on more than one device are able to achieve highly accurate detection rates while reducing the maintenance costs of the models. Depending on the devices and methods that were used for botnet detection, the combined models often produced even greater results than their single-device counterparts.

Lastly, the analysis covered many different feature selection algorithms, powers of the feature sets and anomaly detection and classification algorithms to find the best performing and lowest computational cost models.

Bibliography

[1] F. Wortmann and K. Flüchter, “Internet of Things,” *Business & Information Systems Engineering,* vol. 57, no. 3, pp. 221-224, 2015.

[2] K. L. Lueth, “State of the IoT 2020: 12 billion IoT connections, surpassing non-IoT for the first time,” IoT Analytics, 19 November 2020. [Online]. Available: https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time/.

[3] E. Bertino and N. Islam, “Botnets and Internet of Things Security,” *Computer,* vol. 50, pp. 76-79, 2017.

[4] K. Bhardwaj, J. F. Chung M. and A. Gavrilovska, “Towards IoT DDoS Prevention Using Edge Computing,” in *HotEdge'18*, Boston, 2018.

[5] B. B. Zarpelão, R. S. Miani, C. T. Kawakani and S. C. de Alvarenga, “A survey of intrusion detection in Internet of Things,” *Journal of Network and Computer Applications,* vol. 84, pp. 25-37, 2017.

[6] S. Nõmm and H. Bahşi, “Unsupervised Anomaly Based Botnet Detection in IoT Networks,” ICMLA, 2018.

[7] H. Bahşi, S. Nõmm and F. B. La Torre, “Dimensionality Reduction for Machine Learning Based IoT Botnet,” in *15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Singapore, 2018.

[8] A. K. Jain, R. P. W. Duin and J. Mao, “Statistical pattern recognition: a review,” *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 22, no. 1, pp. 4-37, 2000.

[9] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher and Y. Elovici, “N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders,” *IEEE Pervasive Computing,* vol. 17, no. 3, pp. 12-22, 2018.

[10] L. De Carli, R. Torres, G. Modelo-Howard, A. Tongaonkar and S. Jha, "Botnet protocol inference in the presence of encrypted traffic," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, Atlanta, 2017.

[11] M. N. Sakib and C.-T. Huang, "Using anomaly detection based techniques to detect HTTP-based botnet C&C traffic," in *2016 IEEE International Conference on Communications (ICC)*, Kuala Lumpur, 2016.

[12] M. Stevanovic and J. M. Pedersen, "An efficient flow-based botnet detection using supervised machine learning," in *2014 International Conference on Computing, Networking and Communications (ICNC)*, Honolulu, 2014.

[13] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani and D. Garant, "Botnet detection based on traffic behavior analysis and flow intervals," *Computers & Security*, vol. 39, no. A, pp. 2-16, 2013.

[14] S. Homayoun, M. Ahmadzadeh, S. Hashemi, A. Dehghantanha and R. Khayami, "BoTShark: A Deep Learning Approach for Botnet Traffic Detection," in *Cyber Threat Intelligence*, Springer, Cham, 2018, pp. 137-153.

[15] A. Nogueira, P. Salvador and F. Blessa, "A Botnet Detection System Based on Neural Networks," in *2010 Fifth International Conference on Digital Telecommunications*, Athens, 2010.

[16] R. Doshi, N. Apthorpe and N. Feamster, "Machine Learning DDoS Detection for Consumer Internet of Things Devices," in *2018 IEEE Security and Privacy Workshops (SPW)*, San Francisco, 2018.

[17] D. H. Summerville, K. M. Zach and Y. Chen, "Ultra-lightweight deep packet anomaly detection for Internet of Things devices," in *2015 IEEE 34th International Performance Computing and Communications Conference (IPCCC)*, Nanjing, 2015.

[18] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, D. Breitenbacher, A. Shabtai and Y. Elovici, "detection_of_IoT_botnet_attacks_N_BaIoT Data Set," 19 March 2018. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT.

[19] C. D. McDermott, F. Majdani and A. V. Pertrovski, "Botnet Detection in the Internet of Things using Deep Learning Approaches," in *2018 International Joint Conference on Neural Networks (IJCNN)*, Rio de Janeiro, 2018.

[20] H. Bostani and M. Sheikhan, "Hybrid of anomaly-based and specification-based IDS for Internet of Things using unsupervised OPF based on MapReduce approach," *Computer Communications,* vol. 98, pp. 52-71, 2017.

[21] B. Al-Duwairi, W. Al-Kahla, M. A. AlRefai, Y. Abdelqader, A. Rawash and R. Fahmawi, "SIEM-based detection and mitigation of IoT-botnet DDoS attacks," *International Journal of Electrical & Computer Engineering,* vol. 10, no. 2, pp. 2182-2191, 2020.

[22] M. Özçelik, N. Chalabianloo and G. Gür, "Software-Defined Edge Defense Against IoT-Based DDoS," in *2017 IEEE International Conference on Computer and Information Technology (CIT)*, Helsinki, 2017.

[23] Y. M. Pa Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama and C. Rossow, "IoTPOT: A Novel Honeypot for Revealing Current IoT Threats," *Journal of Information Processing,* vol. 24, no. 3, pp. 522-533, 2016.

[24] Y. Mirsky, T. Doitshman, Y. Elovici and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," in *Network and Distributed System Security Symposium,* San Diego, 2018.

[25] C. C. Aggarwal, Data Mining: The Textbook, New Delhi: Springer International Publishing, 2015.

[26] D. Cai, C. Zhang and X. He, "Unsupervised feature selection for Multi-Cluster data," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, 2010.

[27] L. E. Raileanu and K. Stoffel, "Theoretical Comparison between the Gini Index and Information Gain Criteria," *Annals of Mathematics and Artificial Intelligence,* vol. 41, pp. 77-93, 2004.

[28] A. Banerjee and R. N. Dave, "Validating clusters using the Hopkins statistic," in *2004 IEEE International Conference on Fuzzy Systems,* Budapest, 2004.

[29] Q. Gu, Z. Li and J. Han, "Generalized Fisher Score for Feature Selection," in *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, Barcelona, 2011.

[30] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola and R. C. Williamson, "Estimating Support of a High-Dimensional Distribution," *Neural Computation,* vol. 13, no. 7, pp. 1443-1471, 2001.

[31] V. Vapnik, The Nature of Statistical Learning Theory, New York: Springer, 2000.

[32] Z.-H. Zhou, K. M. Ting and F. T. Liu, "Isolation Forest," in *2008 Eighth IEEE International Conference on Data Mining*, Pisa, 2008.

[33] M. M. Breunig, H.-P. Kriegel, R. T. Ng and J. Sander, "LOF: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, Dallas, 2000.

[34] W. Du and Z. Zhan, "Building decision tree classifier on private data," in *Proceedings of the IEEE international conference on Privacy, security and data mining*, Darlinghurst, 2002.

[35] L. Breiman, "Random Forests," *Machine Learning,* vol. 45, pp. 5-32, 2001.

[36] L. Jiang, Z. Cai, D. Wang and S. Jiang, "Survey of Improving K-Nearest-Neighbor for Classification," in *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*, Haikou, 2007.

[37] D. W. Aha, Lazy Learning, Springer Science & Business Media, 2013.

[38] J. Zhu, S. Rosset, H. Zou and T. Hastie, "Multi-class AdaBoost," *Statistics and Its Interface,* vol. 2, no. 3, pp. 349-360, 2009.

[39] C. Goutte and E. Gaussier, "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation," in *Advances in Information Retrieval*, Berlin, 2005.

[40] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management,* vol. 45, no. 4, pp. 427-437, 2009.

[41] M. Kuhn and K. Johnson, Applied Predictive Modeling, New York: Springer, 2013.

[42] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning, Springer, 2009.

[43] S. Tsimbalist, *Detecting, Classifying and Explaining IoT Botnet Attacks Using Deep Learning Methods Based on Network Data,* Tallinn, 2019.

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[6]

I Rasmus Tomsen

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “In-Depth Feature Selection Analysis for the IoT Based Botnet Attack Detection”, supervised by Sven Nõmm and Hayretdin Bahşi
   1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
   1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

07.05.2021

6 The non-exclusive license is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive license, the non-exclusive license shall not be valid for the period.

Appendix 2 – Source code and full analysis results

Source code and the full analysis results are available at:
https://gitlab.cs.ttu.ee/ratoms/masters-feature-selection